Aalto University
School of Engineering

AALTO UNIVERSITY SCHOOL OF ENGINEERING
Department of Applied Mechanics

Deepak Paudel

Quench Simulation of Superconducting Magnets with Commercial Multi-Physics Software

Thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Technology

Espoo, June 30, 2015

Supervisor: Prof. Dr. Jani Romanoff

Instructor: Dr. Bernhard Auchmann

Aalto University
School of Engineering

Aalto University, P.O. BOX 11000, 00076 AALTO
www.aalto.fi

Abstract of master's thesis

Author	Deepak Paudel
Title of thesis	Quench Simulation of Superconducting Magnets with Commercial Multi-Physics Software.
Degree programme	Mechanical Engineering
Major	Applied Mechanics **Code** K420-3
Thesis supervisor	Prof. Dr. Jani Romanoff
Thesis advisor	Dr. Bernhard Auchmann
Date	30.06.2015 **Number of pages** 60 + 39 **Language** English

Abstract

The simulation of quenches in superconducting magnets is a multiphysics problem of highest complexity. Operated at 1.9 K above absolute zero, the material properties of superconductors and superfluid helium vary by several orders of magnitude over a range of only 10 K. The heat transfer from metal to helium goes through different transfer and boiling regimes as a function of temperature, heat flux, and transferred energy. Electrical, magnetic, thermal, and fluid dynamic effects are intimately coupled, yet live on vastly different time and spatial scales.

While the physical models may be the same in all cases, it is an open debate whether the user should opt for commercial multiphysics software like ANSYS or COMSOL, write customized models based on general purpose network solvers like SPICE, or implement the physics models and numerical solvers entirely in custom software like the QP3, THEA, and ROXIE codes currently in use at the European Organisation for Nuclear Research (CERN). Each approach has its strengths and limitations, some related to performance, others to usability and maintainability, and others again to the flexibility of material parameterizations. In this context the master thesis mainly involves the study of the strengths and limitations of the first approach.

The primary goal of the thesis is to build a 1D numerical model representing a superconducting wire based on existing physical models. An adiabatic model has been constructed, to solve one of the five boundary value problems involved in the quench, both in ANSYS and in COMSOL. The temperature dependent material properties and loads are defined using function tools in COMSOL and by creating look up tables in ANSYS. The models were validated with QP3 and compared in terms of performance, stability and accuracy.

The helium-cooled model is built only in ANSYS. The model solves two of the five boundary value problems simultaneously as a coupled problem. Apart from generic numerical code (transient thermal analysis), a separate algorithm is needed to define the non-linear heat transfer between the metal and the helium. For this ANSYS Parametric Design Language (APDL) scripts are used. During the analysis the ANSYS transient thermal codes are executed several times within a loop. There are three different types of helium cooled models. All models were validated with QP3.

The results obtained from comparisons show that the adiabatic models were able to simulate quenches with the desired accuracy. The adiabatic analysis in the commercial simulation tools is more efficient and stable for various scale of spatial discretization. Similarly, the helium-cooled models are able to simulate quenches with satisfactory accuracy. Nevertheless, the models are not compatible with automatic time stepping method of the simulation environment. The use of fixed time stepping method in the models resulted the coupled analysis in ANSYS to be far more time consuming than in QP3.

Keywords Superconductivity, Quench Simulation, Superconducting Magnets, Numerical Methods

Acknowledgments

Firstly, I would like to thank my university supervisor Jani Romanoff and CERN supervisor Bernhard Auchmann to provide me the thesis opportunity, and for thoroughly checking the thesis document despite of their busy schedule.

Thanks to Michal Maciejewski for helping me during the work as well as thoroughly checking the thesis document. Thanks to my colleagues Ondrej Picha and Scott Rowan for your sincere assistance during the work.

Finally, I would like to thank Arjan Verweij, Emmanuele Ravaioli, Valentina Venturi, Lauri Kortelainen, Markus Bonda, Joel Grognuz, Ulf Friederichs and Pierre-Louis Ruffieux for supporting me during the work.

Table of Contents

Notation

a_{Cu}	cross-section of copper fraction in a strand
a_{FB-I}	heat transfer coefficient, Film boiling-I regime
a_{FB-II}	heat transfer coefficient, Film boiling-II regime
a_{He}	total contact surface area between helium and conductor
a_{kap}	heat transfer coefficient, Kapitza regime
a_{NB}	heat transfer coefficient, Nucleate Boiling regime
a_{NC}	heat transfer coefficient, Natural Convection regime
a_{wire}	cross-section of a strand
B_c	critical magnetic field
B_{c20}	intercepts of critical surface along magnetic field axis
$c_{p,cu}$	specific heat capacity of copper
$c_{p,model}$	equivalent heat capacity of a model
$c_{p,Nb-Ti}$	specific heat capacity of Nb-Ti superconductor
$c_{p,strand}$	equivalent heat capacity of a strand
d_{Cu}	equivalent diameter of copper fraction
d_{wire}	diameter of a strand
f	copper to superconductor ratio in a Strand
f_{Cu}	fraction of copper volume in a strand
h_f	heat transfer coefficient
h_{Joule}	joule heat generated per unit length in a strand
h_{He}	rate of heat transfer to the liquid helium per unit area.
I_c	critical current
I_{NC}	current in the copper matrix
J_c	critical current density
k	thermal conductivity
L	length of a strand
S	heat generation load
T_b	bath temperature
T_c	critical temperature
T_{c0}	intercepts of critical surface along temperature axis
T_{cs}	current sharing temperature
T_{He}	temperature of the liquid helium
T_{op}	operational temperature
T_{peak}	hot-spot temperature
T_s	temperature of a strand
T_λ	lambda temperature
n_e	empirical power constant
Q_{kap}	heat flux limit, Kapitza regime
Q_{NC}	heat flux limit, Natural Convection regime
Q_{NB}	heat flux limit, Nucleate Boiling regime
V_{He}	total volume of helium around a strand
ρ_{Cu}	specific resistivity of copper
ρ_{He}	density of helium

1. INTRODUCTION

The main objective of the thesis is to build a quench simulation model of superconducting wire, based on Finite Element Method (FEM), using general purpose commercial Multi-Physics software packages, and to validate and compare the model with CERN in house software that was specifically built for this purpose. In this chapter, we introduce some basic concepts of quenches and quench simulation.

1.1 Superconductivity

Superconductivity is a unique property of certain materials that, when they are cooled down to very low temperatures, exhibit zero resistivity. The very temperature below which the conductor starts behaving like a superconductor is called critical temperature T_c. Besides critical temperature there are two other parameters (i) critical current density J_c and (ii) critical magnetic field B_c that determine whether the metal is in superconducting state or normal conducting state. Assuming these parameters are the Cartesian coordinates, the region where metal behaves as superconducting, and normal conducting outside the very region, can be plotted. Figure 1 shows the critical surface of Nb-Ti alloy. Nb-Ti is commercially available and widely used alloy in superconducting applications [1].

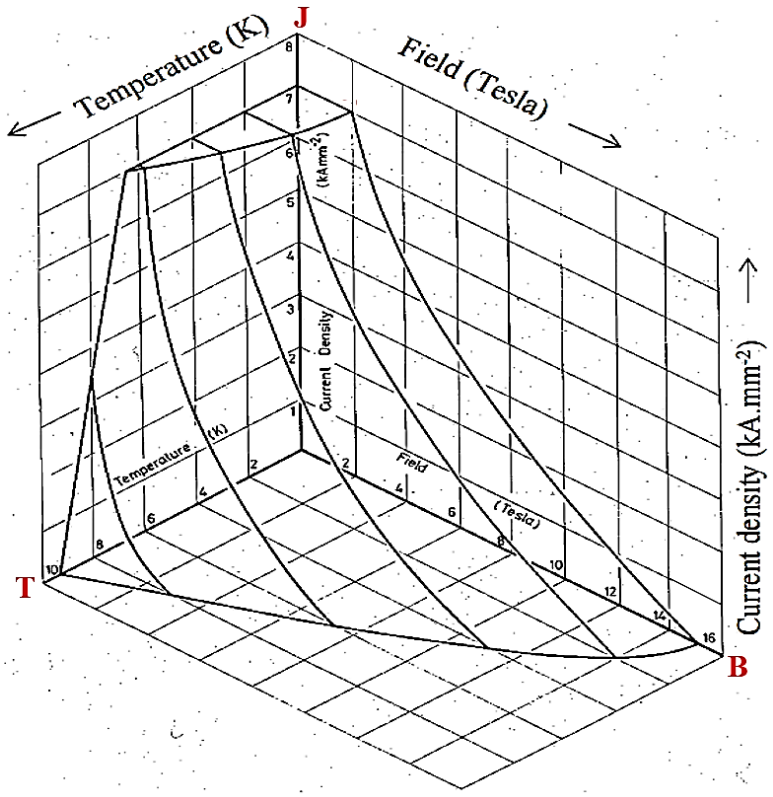


Figure 1 Critical surface for Nb-Ti [2].

1.2 Superconducting Magnets

To benefit from high efficiency and reduced size, superconducting magnets are used in various applications such as magnets for nuclear fusion, Magnetic Resonance Imaging (MRI), MagLev Magnetic Levitation (MagLev), and accelerator magnets [3 and 4].

The Large Hadron Collider (LHC) is a 27-kilometer-long chain of mostly superconducting magnets which includes 1232 superconducting dipole magnets of length 15 metres to bend the beam of particles, 392 superconducting quadruple magnets of length 2.5-7 metres to focus the beam, and other more to steer the beam just before the collision[5].

The LHC ring is depicted in Figure 2. Two separate beams of particles are injected in opposite direction and the collision of the particles takes place at particle detectors (A Toroidal LHC Apparatus (ATLAS), Compact Muon Solenoid (CMS), A Large Ion Collider Experiment (ALICE), and Large Hadron Collider beauty (LHCb)) where particle dynamics is studied. The Radiofrequency (RF) cavities generate electromagnetic fields which produce forces to accelerate the beam of particles once per turn. The cleaning sections in the ring consist of structures called collimators. They scrape off off-center particles in order to protect the accelerator against beam losses.

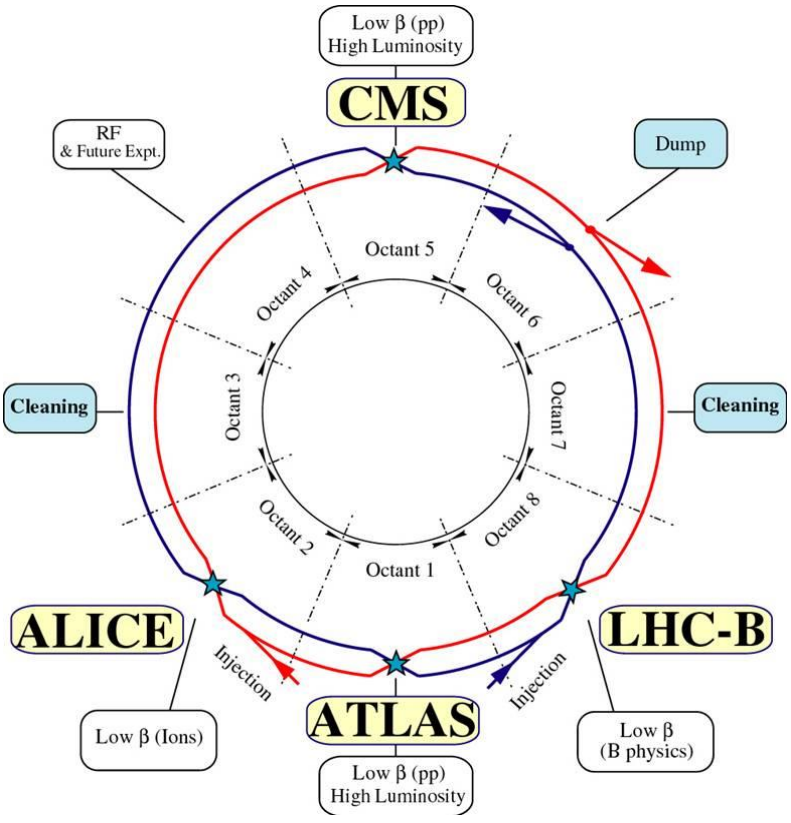


Figure 2 The Large Hadron Collider [5].

Superconducting wire also known as superconducting strand is the basic element of the superconducting magnet coils. Figure 3e shows the Nb-Ti type superconducting strand where tiny (~7 µm) Nb-Ti filaments are bundled together in a number of hexagons and embedded in a copper matrix [1].

High-current Rutherford-type cable (Figure 3c) are used in LHC superconducting magnets. Such cables are flattened helices usually comprising 20 to 40 strands. In order to reduce cable eddy currents the strands are fully transposed with cable twist pitch of 6-8 times the cable width which in turn improves the field homogeneity during the ramp [6].

The coil of LHC dipole magnets consists of two layers of Rutherford-type cable. The cables are insulated to prevent shorts between the adjacent turns. The superconducting state is obtained by immersing the magnet in a bath of liquid helium, which acts as a heat sink [7]. The cross-sections of the coil along with the beam tube and the supporting structure is shown in the Figure 3b.

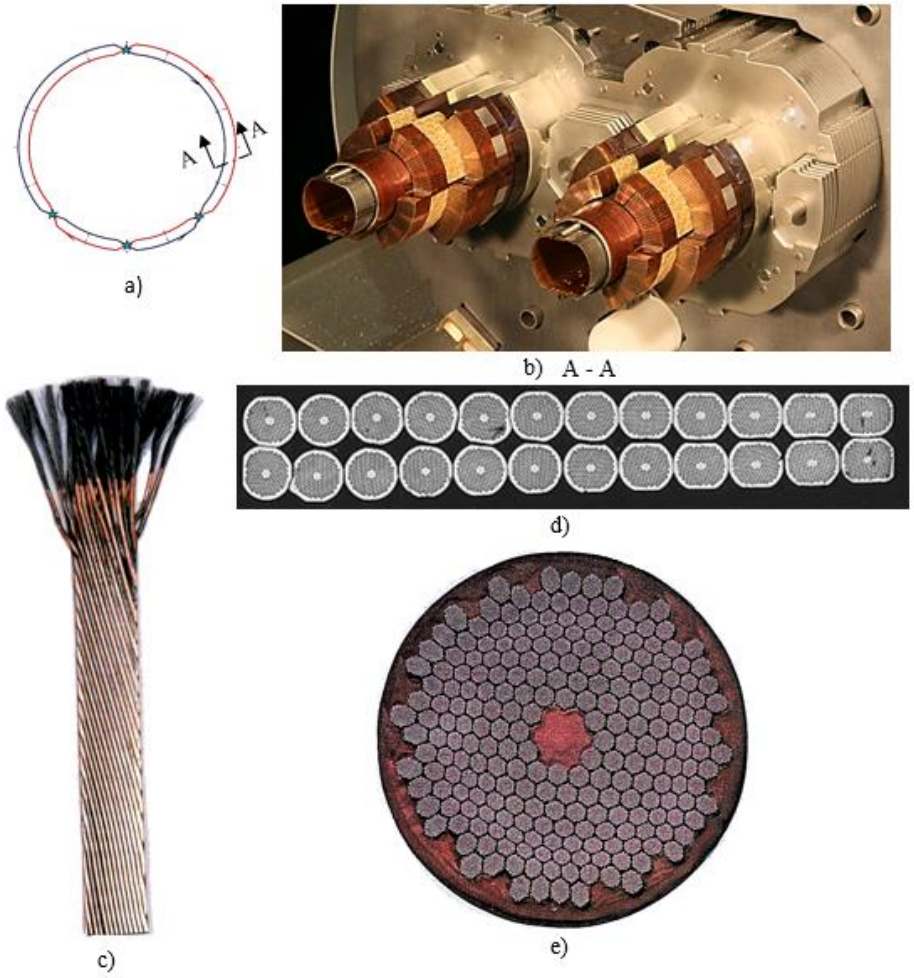


Figure 3 a) Symbolic representation of the LHC ring. b) Cross-section of coil of superconducting dipole magnet. c) Rutherford-type cable. d) Cross-section of Rutherford-type cable. e) Cross-section of a superconducting strand [1].

1.3 Quench Simulation

The transition of a conductor from the superconducting to the normalconducting state is called quench. Typically a quench occurs because of sudden movement and friction-induced heating, beam-induced losses and/or ramp-rate induced eddy-current losses; see Appendix 4.

In superconducting state, the resistivity of Nb-Ti filaments is zero and the entire current flows through the filaments. During a quench, as the resistivity of the normalconducting filaments is orders of magnitude higher than that of copper, the current deviates into the copper matrix and generates Joule heating. The heat generated is taken away by liquid helium. However if the rate of heat generation in the copper matrix persists to be greater than the rate of heat transfer to the liquid helium the temperature of the conductor increases. Through heat conduction it is transferred along the longitudinal direction of the strand and in the transverse direction to the other strands thus successively raising the temperature beyond the critical temperature and propagating the quench.

During the process the initial quench spot (so-called hot-spot) is exposed to high temperature. The high temperature can lead to a meltdown of the insulation or even the copper matrix and damage the magnet permanently. In order to protect the magnet, a quench needs to be detected in time and protection schemes need to be implemented. For this, a proper understanding of the quench behaviour is essential.

1.4 Background to the research

At CERN, research is being carried out to study the nature of elementary particles and forces between them. The particles like protons are accelerated to nearly the speed of light, in the LHC ring. When the beam of particles reaches the desired energy it is made to collide with another beam travelling in the opposite direction. The experiments probe the basic nature of the most fundamental particles.

Superconducting magnets in LHC are operated at cryogenic temperature; 1.9 K. Operating magnets at such a low temperature, without a proper protection measures poses a threat of magnet being permanently damaged by quenching. Simulation of quenches can help us understand probable problems and how to resolve them before they occur in real world situations.

CERN has its own in-house pieces of software for quench simulation such as Quench Protection (QP3) [8], Thermal, Hydraulic and Electric Analysis (THEA) [9] and Routine for the Optimization of magnet X-sections, Inverse field calculation and End design (ROXIE) [10]. QP3 is a FORTRAN code based on a 1D thermal network. It simulates the quench behaviour of a longitudinally discretised conductor by an algorithm that goes through different loops and iterations in order to solve the coupled and highly non-linear problem. THEA performs 1D/2D static and transient multi-physics analysis of generic superconducting cable by thermal and hydraulic networks. ROXIE is a 2D/3D simulation software for the design and optimization of LHC superconducting magnets, featuring a magnet-level basic thermal network solver.

Similarly the use of multiphysics software like ANSYS and COMSOL [11, 12 and 13] and network solvers like Simulation Program with Integrated Circuit Emphasis (SPICE) [14] has been reported. ANSYS and COMSOL are examples of widely used Finite Element Analysis (FEA) tools, whereas SPICE is a general purpose, open source network solver.

The Simulation of Transient Effects in Accelerator Magnets (STEAM) project, recently started at CERN, aims to study the stability and performance of different approaches to quench simulation in order to answer the following questions.

1. ***Can commercial multiphysics software deal with the complex physics models related to superconductivity and superfluidity?***

Commercial software has evolved over the decades and provides fast and accurate solutions for a wide range of physical problems. However the codes may not include features to deal with very specific problems such as superconductivity and superfluidity. Custom made software at CERN was built specifically to solve such problems. Nevertheless in terms of performance they

might not be as efficient as compared to commercial software. The strengths and limitations of both type of simulation environment should be compared by means of relevant benchmark.

2. ***Can all relevant physical effects be coupled?***

In order to understand the quench behaviour of a magnet all involved physical phenomena have to be treated as a coupled multiphysics problem. Most popular commercial software packages have already incorporated the multiphysics features in their codes. For example, ANSYS has two different methods to solve the coupled field problems.

I. Direct Method, where a coupled-field element is used that contains all necessary degrees of freedom and the corresponding stiffness matrix and the load vector takes into account all necessary terms.

II. Load Transfer Methods, which involve two or more analyses coupled by applying results from one analysis as a load to another analysis.

Can these methods couple all the involved physical effects and if yes how efficient they can be?

3. ***Can time stepping and spatial discretization be adaptive per physical model and geometric subdomain?***

Physical problems encountered during quenching exhibit different spatial and temporal scales of integration constants, which are not uniformly distributed in the space-time domain. In order to take advantage of the efficient numerical discretization, it is recommended to use adaptive methods per physical and/or geometrical subdomains. Are there any straight forward solutions or predefined functions in commercial software to use such features?

4. ***How does commercial multiphysics software compare to custom-made software in terms of stability, accuracy and performance for our particular problem?***

For now it is still unknown whether a commercial multiphysics software can simulate the specific problems like quench taking into account all involved physical effects. If yes how efficient would they be in terms of performance compared to custom-made software?

The thesis project is a part of the STEAM project which studies the coupling of different commercial and in-house programs in a co-simulation framework. The result of the thesis project shall give answers to questions 1 and 4 and contribute in providing necessary information to prolong the research in finding answers to question 2 and 3.

2. QUENCH PROBLEM

Theoretical models that explain the physics behind the quench are already formulated and documented in literature, e.g., [1, 6 and 7]. The simulation of quenches is a multiphysics problem. Large magnets have multi-scale components from sizes as small as ~7 μm to 15 m in length, with highly non-linear material properties that vary by several orders of magnitude over a range of only 10 K. Moreover, various physical problems are entangled with each other [15 and 16]. The coupled physical boundary value problems are:

A) The electrical problem: non-linear current-voltage characteristics of the superconductor; non-linear dependency of the conductor resistivity on magnetic field, temperature.
B) The magnetic problem: non-linear inductance and eddy-current effects inside the coil and in other structural elements.
C) The heat transfer from solid to helium: the heat transfer from the conductor to the helium goes through different transfer and boiling regimes as a function of temperature, heat flux, and transferred energy.
D) The thermal problem in solids: Joule losses in conductor, temperature dependent thermal conductivity and heat capacity.
E) The thermal and fluid-dynamic problem of helium: temperature-dependent viscosity, heat capacity, density, and thermal conductivity.

Among above five different boundary value problems the thesis project covers C and D. D depends upon the current and magnetic field distribution obtained from A and B. Similarly the heat transfer between the conductor and helium, C strongly depends upon E.

To make the problem simple the current and magnetic field will be considered to remain constant throughout the analysis. As a first step only D will be solved, i.e, the adiabatic model presented in Chapter 4.2. Then C and D will be solved as a coupled problem, i.e, the helium-cooled model presented in Chapter 4.3.

Since C and D are strongly depended on A, B and E. It is essential to understand the relation between these problems. In the following sections we present the related topics. In Sections 2.1 and 2.2 we elaborate C and D respectively.

Copper Resistivity

The specific resistivity of the copper depends upon the temperature *T* and magnetic field *B;* see Appendix 5. Residual Resistivity Ratio (RRR) is the ratio of the resistivity of the material at room temperature and the cryogenic temperature (practically defined as 4 K) [4]. It is

determined by the purity of the copper. The higher the RRR value, the lower will be the resistivity.

Critical current I_c(B,T)

The critical current in the superconductor depends upon the magnetic field *B* and the temperature *T*. The critical current for Nb-Ti superconductor is fitted by [17]

$$I_c(B,T) = (C_1 + C_2 \cdot B)(1 - \frac{T}{T_c(B)}). \qquad [A] \qquad (2.1)$$

The magnetic field direction is perpendicular to the current. The constants C_1 = 3449 A and C_2 = -257 AT^{-1} for LHC type 01 cable are given in and T_c (*B*) is the critical temperature defined as [4]

$$T_c\,(B) = T_{c0}(1 - \frac{B}{B_{c20}})^{0.59}, \qquad [K] \qquad (2.2)$$

where T_{c0} = 9.2 K and B_{c20} = 14.5 T for Nb-Ti type superconductor; see Figure 5.

Current in the Copper-Matrix

Suppose the bath temperature in the magnet is T_b. Using (2.1) and (2.2) the critical current at bath temperature is

$$I_c(B,T_b) = (C_1 + C_2 \cdot B)(1 - \frac{T_b}{T_c(B)}), \qquad [A] \qquad (2.3)$$

using (2.1) and (2.3) critical current at temperature *T* is

$$I_c(B,T) = I_c(B,T_b)\frac{T_c - T}{T_c - T_b}, \quad T < T_c. \qquad [A] \qquad (2.4)$$

As one can notice from (2.4) when the temperature *T* in the conductor increases, the critical current I_c (*B, T*) decreases. When it drops below the transport current *I*, the excess current starts to commute to the copper matrix. The temperature at which the current starts to commute to copper matrix is known as current sharing temperature T_{cs}. The current sharing temperature for the magnetic field *B* and current *I* can be calculated as [10]

$$T_{cs}(I,B) = T_c(B)\left(1 - \frac{I}{(C_1 + C_2 B)}\right). \qquad [K] \qquad (2.5)$$

A plot of critical surface showing the working line, T_{c0}, B_{c20}, and T_{cs} is shown in Figure 4.

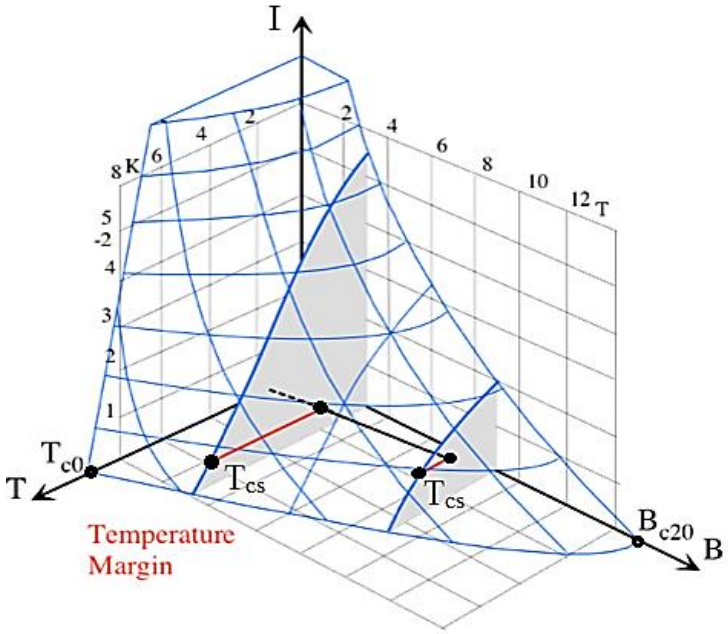


Figure 4 Critical surface of Nb-Ti superconductor showing T_{c0}, B_{c0}, and T_{cs} for two different working points along the load line [2].

The current in the copper matrix is equal to the transport current when the temperature of the conductor rises beyond the critical temperature T_c. Following the above assumptions the current in the copper matrix can be calculated using the empirical formula [10].

$$I_{NC}(T) = \begin{cases} 0 & \text{for} \quad T < T_{cs} \\ I - I_c & \text{for} \quad T_{cs} < T < T_c. \\ I & \text{for} \quad T > T_c \end{cases} \qquad [\text{A}] \qquad (2.6)$$

Ohmic heating

Ohmic heating or Joule heating is the process in which heat is generated in a conductor as the current passes through it. Assuming $\vec{J}$ is the current density vector at a point in the conductor where the temperature- and the field-dependent specific resistivity is $\rho\,(T, B)$. The Joule heating at the very point can be expressed as [10]

$$h_{\text{Joule}}(T, B) = \rho(T, B)\vec{J} \cdot \vec{J}. \qquad [\text{Wm}^{-3}] \qquad (2.7)$$

Assuming a Nb-Ti superconducting wire to be a one dimensional line with constant copper cross-section area of a_{cu}, the Joule heat generated per unit length of the wire can be calculated using (2.6) and (2.7)[10],

$$h_{\text{Joule}}(T, B) = \begin{cases} 0 & \text{for} \quad T < T_{cs} \\ \rho_{cu}(T, B)\frac{(I - I_c)^2}{a_{cu}{}^2} & \text{for} \quad T_{cs} < T < T_c, \\ \rho_{cu}(T, B)\frac{I^2}{a_{cu}{}^2} & \text{for} \quad T > T_c \end{cases} \qquad [\text{Wm}^{-3}] \qquad (2.8)$$

where $\rho_{cu}(T, B)$ is the resistivity of copper, T_{cs} and T_c are the current-sharing temperature and critical temperature of the superconductor respectively.

Helium thermal properties

Helium is the only substance that remains liquid near the absolute zero temperature and does not solidify below the pressure of 2.5 MPa. It has two liquid phases separated by a lambda line, $T_\lambda \sim 2.16$ K; see Figure 5. Left to the lambda line it exists as superfluid liquid (He-II), i.e. it has no measurable flow resistance (viscosity), and right to the line it exists as a normal fluid (He-I) with measurable viscosity comparable to air [1, 3, 19, and 20].

The operating condition of the LHC superconducting magnets is indicated by a black dotted line in Figure 5. The pressure remains near constant and He-II and He-I phases of helium are used for cooling. As the temperature increases from 1.9 K to 4.5 K, helium goes through phase changes (from He-II to He-I, and gaseous He) with highly non-linear behavior. The density of the liquid decreases and there is a significant change in the thermal capacity and the conductivity of the liquid; see Appendix 10 and 11.

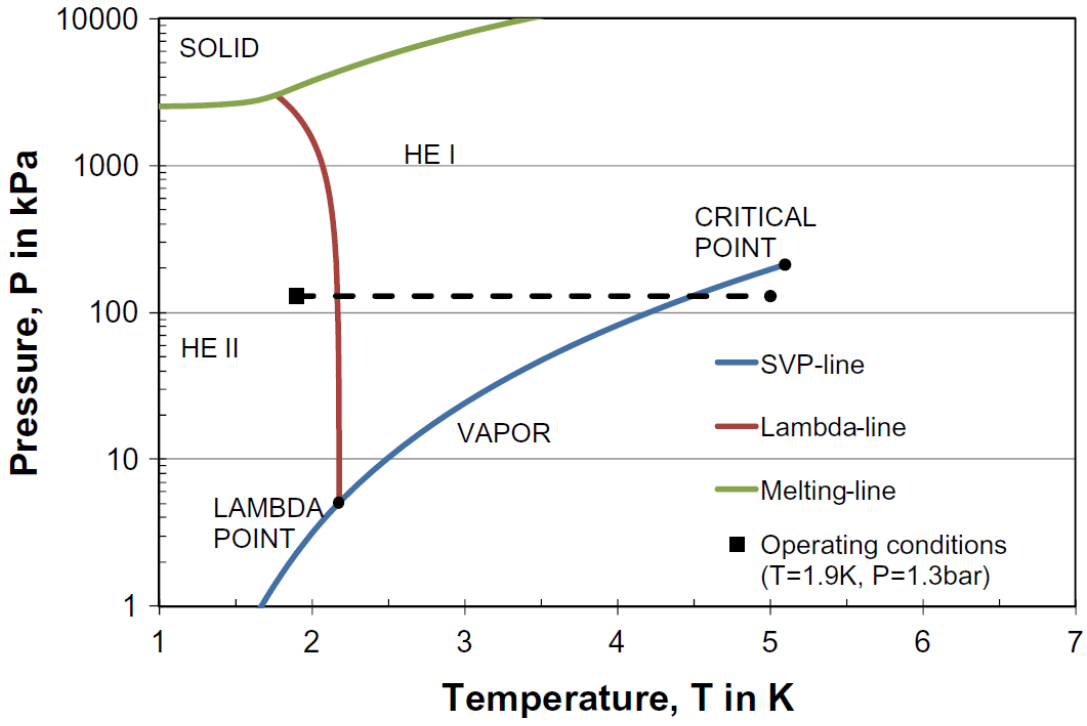


Figure 5 Helium Phase Diagram [19]

2.1 Heat transfer from solid to helium

The heat transfer between the conductor and helium can be explained by assuming altogether five different regimes and two different phases; see Figure 6.

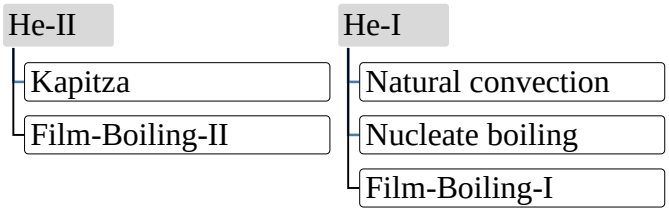


Figure 6 Different assumed regimes to explain the non-linear heat transfer between conductor and helium.

He-II has unique cooling properties. The specific heat capacity and the thermal conductivity in this phase are orders of magnitude higher than that of the electrical conductor. There are two regimes explaining the heat transfer between He-II and the conductor.

2.1.1 Kapitza regime

Below a certain limit of heat flux, Q_{kap} the cooling is dominated by Kapitza conductance. The phenomenon provides the most effective heat flow. In general if two domains are in contact there is a smooth temperature change in the interface as we go from one domain to another. However between the conductor and He-II there exists sizable temperature jump. Details can be found in [1]. The heat flow in this regime is given as

$$h(T_s, T_{He}) = a_{kap}\left(T_s^{n_{kap}} - T_{He}^{n_{kap}}\right), \qquad [Wm^{-2}] \qquad (2.9)$$

where T_s is the temperature of the conductor, T_{He} is the temperature of the helium. a_{kap} and n_{kap} are the heat transfer coefficient (HTC) and the power coefficient, respectively. Typical values are 200 Wm^{-2} K^{nkap} and 4, respectively. The heat flux limit Q_{kap} typically used is 35000 Wm^{-2}.

2.1.2 Film boiling-II regime

Beyond the limit of heat flux Q_{kap} a thin layer of helium vapor and/or liquid He-I is formed between the surface of the conductor and the He-II. It prevents He-II to be in direct contact with the conductor surface. The heat transfer is much less effective because of the low thermal conductivity of the layer formed. If the heat flux drops below the limit, the cooling behavior falls back to the Kapitza regime. The heat flow is expressed as

$$h(T_s, T_{He}) = a_{FB\text{-}II}(T_s - T_{He}). \qquad [Wm^{-2}] \qquad (2.10)$$

The typical value of $a_{FB\text{-}II}$ is 250 $Wm^{-2}K^{-1}$.

He-I is a normal liquid with small thermal conductivity and large specific heat capacity. This suggests that the heat transfer in this state is governed by convection and boiling rather than by conduction. The He-I cooling can be modeled by three different regimes.

2.1.3 Natural Convection regime

Below a few Wm^{-2} of heat flux there is no phase change and cooling is assumed to be purely by convection,

$$h(T_s, T_{He}) = a_{NC}(T_s - T_{He}). \qquad [Wm^{-2}] \qquad (2.11)$$

The typical value of a_{NC} is 500 $Wm^{-2}K^{-1}$. The regime is assumed to continue until the heat flux limit of Q_{NC}. The typical value of Q_{NC} is 10 Wm^{-2}.

2.1.4 Nucleate Boiling regime

As the heat flux crosses the limit of pure convection, helium vapor is assumed to form at preferred sites on the surface of the conductor. On increasing the heat flux the nucleation sites get fully activated and form bubbles. The rate of bubble growth increases with increase in the heat flux. As the bubbles detach from the surface, the cold liquid rushes down to cool the surface. The following expression gives the amount of heat flow,

$$h(T_s, T_{He}) = a_{NB}(T_s - T_{He})^{2.5}. \qquad [Wm^{-2}] \qquad (2.12)$$

The typical value of a_{NB} is 50,000 $Wm^{-2}K^{-2.5}$. The heat flux limit Q_{NB}, typically used for this regime is 10,000 Wm^{-2}.

2.1.5 Film-Boiling regime

At still higher heat fluxes, the rate of bubble detachment increases. The bubbles become unstable and they form a layer of helium vapor at the interface preventing He-I from being in direct contact with the conductor. The heat flow is given by

$$h(T_s, T_{He}) = a_{FB\text{-}I}(T_s - T_{He}). \qquad [Wm^{-2}] \qquad (2.13)$$

The typical value of $a_{FB\text{-}I}$ is 250 $Wm^{-2}K^{-1}$. Upon decrease in the heat flux, there is the possibility to fall back to nucleate boiling and natural convection regime through a hysteretic effect. The hysteresis is not taken into account in our models.

From the above assumptions an empirical formula to calculate the heat flow between the conductor and the helium phases can be constructed,

$$h_{He}(T_s, T_{He}) = \begin{cases} a_{kap}\left(T_s^{n_{kap}} - T_{He}^{n_{kap}}\right) & \text{for} \quad T_{He} < T_\lambda \quad \& \quad h_{He} < Q_{kap} \\ a_{FB\text{-}II}(T_s - T_{He}) & \text{for} \quad T_{He} < T_\lambda \quad \& \quad h_{He} \geq Q_{kap} \\ a_{NC}(T_s - T_{He}) & \text{for} \quad T_{He} \geq T_\lambda \quad \& \quad h_{He} < Q_{NC} \\ a_{NB}(T_s - T_{He})^{2.5} & \text{for} \quad T_{He} > T_\lambda \quad \& \quad h_{He} > Q_{NC} \\ a_{FB\text{-}I}(T_s - T_{He}) & \text{for} \quad T_{He} > T_\lambda \quad \& \quad h_{He} \geq Q_{NB} \end{cases} . \quad [Wm^{-2}] \quad (2.14)$$

2.2 Thermal problem in solids

The main elements of the thermal problem in the superconductor are Joule heating and thermal conduction. Joule heating depends upon temperature and magnetic field (2.7) and the thermal properties are temperature-dependent; see Appendix 8 and 9.

Let $T(\boldsymbol{X}, t)$ the temperature at time t in point $\boldsymbol{X}$ of a conductor domain Ω with boundary, Γ; see Figure 7. The change in the internal energy per unit time, at point $\boldsymbol{X}$, inside the domain can be given as [21]

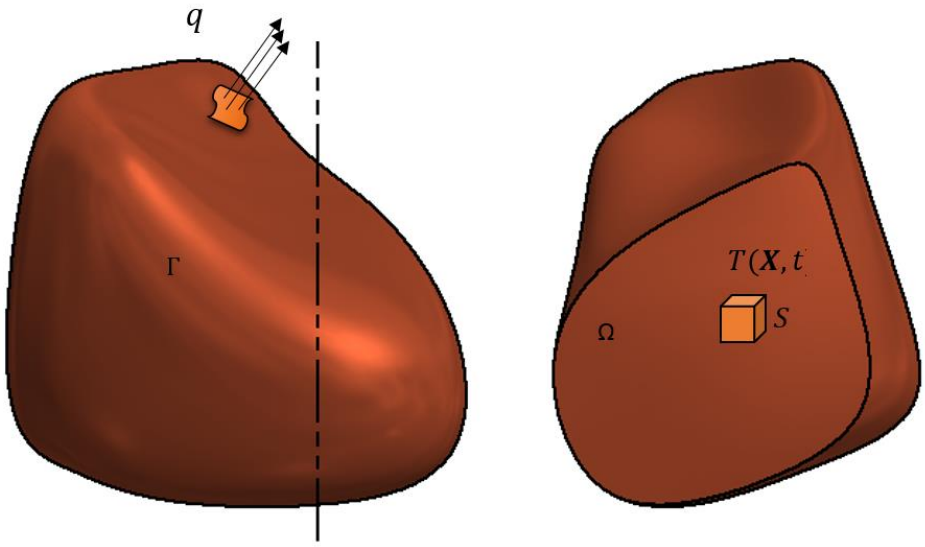


Figure 7 Arbitrary domain subjected to internal heat source and surface flux distribution.

$$\frac{\partial U}{\partial t} = \rho c_p \frac{\partial T}{\partial t}, \tag{2.15}$$

where ρ is the mass density and is assumed to be constant. c_p is the temperature-dependent specific heat capacity. The law of energy conservation states that the net energy flowing into the domain per unit time must be equal to the change of internal energy inside the domain,

$$\int_\Omega S d\Omega - \int_\Gamma \mathbf{n} \cdot \mathbf{q} d\Gamma = \int_\Omega \rho c_p \frac{\partial T}{\partial t} d\Omega, \tag{2.16}$$

where S is the temperature- and the field-dependent heat source distributed throughout the domain. $\mathbf{n}$ is the normal vector and $\boldsymbol{q}$ the heat flux on the boundary. Using Gauss theorem on the integral involving the surface flux,

$$\int_\Omega S d\Omega - \int_\Omega \boldsymbol{\nabla} \cdot \mathbf{q} d\Omega = \int_\Omega \rho c_p \frac{\partial T}{\partial t} d\Omega, \tag{2.17}$$

and hence

$$\int_{\Omega}[S - \boldsymbol{\nabla} \cdot \mathbf{q} - \rho c_{\mathrm{p}} \frac{\partial T}{\partial t}]\mathrm{d}\Omega = 0. \tag{2.18}$$

Conservation of energy requires that (2.18) be true for all subdomains within the domain. This yields

$$S - \boldsymbol{\nabla} \cdot \mathbf{q} - \rho c_{\mathrm{p}} \frac{\partial T}{\partial t} = 0. \tag{2.19}$$

Fourier's law gives the relation between heat flux and temperature gradient, i.e,

$$\mathbf{q} = -k\boldsymbol{\nabla}\mathrm{T}, \tag{2.20}$$

where *k* is the temperature- and the field-dependent thermal conductivity. Using (2.19) and (2.20) we get the second order heat balance equation [21],

$$\rho c_{\mathrm{p}} \frac{\partial T}{\partial t} = \boldsymbol{\nabla} \cdot (k\boldsymbol{\nabla}\mathrm{T}) + S. \tag{2.21}$$

It is a governing differential equation to compute the thermal effects of the quench problem.

3. NUMERICAL SOLUTION

From a numerical-analysis point of view, quench simulations is one of the most interesting, challenging and demanding subject in the superconducting magnet technology. Since the 1970's various computational methods have been adopted to understand the quench behaviour and successful 1D and 2D results have been reported for the strands and windings [11-13 and 22-24]. However the simulation of large superconducting magnets is still a challenging task even with today's powerful computers and advanced computational methods. We present four main aspects to be taken into consideration while solving the quench problem numerically.

a) Transient problem

Quench simulation is a transient thermal problem. The simulation model seeks to solve the temperature profile of the coil and its derivatives as a function of time. The accurate evaluation requires the calculation of the temperature profile with smallest time steps in the order of a micro-second.

b) Complex geometry of the coil

The geometry of the LHC superconducting magnet coil is designed based on a delicate mathematical formulation [10]. The geometry construction for the numerical analysis using such mathematical formulation is difficult. Considering the geometry to be a simple (cylindrical/rectangular) shape does not simulate the problem with required accuracy. An alternative approach is to import the geometry from a Computer Aided Design (CAD) system into the simulation environment and simplify it for the analysis. An accurate model requires construction of the actual up to 15 m long coil geometry with finite element discretization in the scale of a millimeter.

c) Non-linear heat flux in the boundaries

The heat flux between the conductor and helium interface depends upon the temperature of the conductor and the helium in contact, as well as on the heat-flux and integrated transferred energy. A proper helium cooled model would take into account the phase change of the helium and heat transfer through different boiling regimes; see Section 2.1.

d) Inhomogeneous and non-linear material properties

The coil is made of superconductor, copper and the insulation material, each having material properties which exhibit a non-linear dependency on the temperature and the magnetic field. The finite element method can easily take into account in-homogeneous material properties in the calculation of the stiffness matrix. However, the non-linear problem usually involves some

form of iteration or drastically reduced time steps, and the process can be computationally expensive. An efficient model would use adaptive temporal and spatial integration methods in order to reduce the computation time.

The transient problem involving complex geometries, non-linear and inhomogeneous material properties are discussed in the literature [25 and 26]. The popular commercial software packages such as ANSYS, COMSOL, and ABAQUS have user friendly graphical user interfaces to construct and solve such problems. However, modelling of the helium cooling in the coil is still a challenging task. It is difficult to formulate the problem with coefficients depending on two temperatures, heat flux and integrated heat. The traditional approach would be to solve the temperature degree of freedom at each time step using numerical codes and to introduce the cooling effect, based on the evaluated temperature degree of freedom of the current time step, as a boundary load for the next time step. In other words coupling between the generic numerical codes of the simulation environment with custom made cooling algorithm. The methods are presented in Section 4.3.

In Chapter 2 we formulated the quench problem by deriving the governing heat diffusion equation (2.21), the equation of joule heat generation (2.8) and the equation explaining heat transfer between the helium and the superconducting wire (2.14). In Chapter 4 we will present a numerical model of the superconducting wire, based on these equations, using commercial software packages. These software packages are based on FEM. In this method the problem domain is discretized into a number of small subdomains. The material constants, loads and boundaries in each domain are expressed in matrix form. The equilibrium condition (balance between internal forces corresponding to material properties and applied loads and boundaries) is found by matrix manipulation. In the following sections (3.1 to 3.4), starting from the governing heat diffusion equation (2.21), we present how such matrices are formed and computed in the commercial simulation packages. The sections covers the basic theory of the FEM, with a 1D superconducting wire as example. Furthermore, the sections 3.5 and 3.6 presents the numerical theory adopted by commercial software packages to solve the transient and non-linear material problems respectively. In order to build an accurate, stable and efficient numerical model it is essential to thoroughly understand the theoretical and practical information provided in this chapter.

The steps involved in the FEM can be summarized as following [26 and 27]:

- Formulation of the problem in differential form along with boundary conditions. The expression known as strong from.
- Transformation of the differential equation into weighted integral equations, integration by parts to reduce the order of differential term in the integral equation and integration of natural boundary condition in the integral form. The final expression known as the weak form.
- Discretization of the domain into finite elements and approximation of the solution with unknown degrees of freedom and shape functions.
- Expression of the equilibrium condition of all subdomains in matrix forms, using shape functions and weighting functions, and solution of the unknown values.

3.1 The strong form

The strong form of the problem is the governing differential equation along with the necessary boundary conditions. For a 1D superconducting strand the heat balance equation (2.21) reduces to

$$\rho c_{\mathrm{p}} \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right) + S, \quad x \in \Omega = [0,\ L],\ t \in \psi = [0, t], \tag{3.1}$$

where x is the spatial coordinate, L the length of the superconducting wire, and t the temporal coordinate. The boundary condition is the restriction on the field variable or its derivatives on the boundary. In our case temperature, $T(x, t)$ is the field variable and its first derivative is proportional to heat flux. The restriction the on field variable is called Dirichlet or essential boundary condition and the restriction on the first derivative is called Neumann condition or natural boundary condition. The spatial boundaries are the two ends of the wire. We assume zero heat flux on both ends i.e. homogeneous Newman boundary conditions,

$$\vec{\mathrm{n}} \cdot \vec{\mathrm{q}} = \vec{\mathrm{n}} \cdot k \frac{\partial T}{\partial x} \vec{e} = 0, \qquad x \in \partial\Omega = \{0, L\}, \tag{3.2}$$

where $\vec{\mathrm{n}}$ is the outward pointing normal vector and $\vec{e}$ is the direction of increasing x-coordinate. The initial condition for the transient problem is

$$T(x, t) = g(x) \qquad \text{for} \qquad t = 0, \tag{3.3}$$

where $g(x)$ represents the temperature distribution on the wire at time $t = 0$.

3.2 The weak form

While solving differential equations analytically, a suitable function is introduced in the differential equation and it is checked whether the function is a solution or not. To derive FEM, a different approach is applied. Rather than searching for a solution fulfilling the above strong formulation, it is required that the equation is fulfilled "in a weighted sense" for appropriately chosen weighting functions $\omega(x)$. To facilitate the process, the strong form is changed to weak form.

Multiplying (3.1) by a weighting function $\omega(x)$, rearranging and integrating over the spatial domain, we get

$$\int_\Omega [\rho c_\mathrm{p} \frac{\partial T}{\partial t} \omega - \frac{\partial}{\partial x}\left(k \frac{\partial T}{\partial x}\right) \omega - S\omega] \mathrm{d}\Omega = 0. \tag{3.4}$$

The higher the order of derivative term in the problem formulation the higher has to be the order of polynomial functions of the approximation function. Integrating by parts reduces the order of the derivative term in the integral form. We integrate by parts in the second term on the left hand side of (3.4) to get

$$\int_\Omega \rho c_\mathrm{p} \frac{\partial T}{\partial t} \omega \mathrm{d}\Omega + \int_\Omega k \frac{\partial T}{\partial x} \frac{\partial \omega}{\partial x} \mathrm{d}\Omega - \int_\Omega \frac{\partial}{\partial x}(k \frac{\partial T}{\partial x} \omega) \mathrm{d}\Omega - \int_\Omega S\omega \mathrm{d}\Omega = 0. \tag{3.5}$$

By applying the fundamental theorem of calculus in the third term on the left hand side of (3.5) we obtain

$$\int_\Omega \rho c_\mathrm{p} \frac{\partial T}{\partial t} \omega \mathrm{d}\Omega + \int_\Omega k \frac{\partial T}{\partial x} \frac{\partial \omega}{\partial x} \mathrm{d}\Omega - \int_\Gamma \vec{\mathrm{n}} \cdot k \frac{\partial T}{\partial x} \vec{e} \omega \mathrm{d}\Gamma - \int_\Omega S\omega \mathrm{d}\Omega = 0. \tag{3.6}$$

Next, using the spatial boundary condition (3.2) in (3.6) and re-arranging the terms, the weak formulation of the problem reads

$$\int_\Omega \rho c_\mathrm{p} \frac{\partial T}{\partial t} \omega \mathrm{d}\Omega + \int_\Omega k \frac{\partial T}{\partial x} \frac{\partial \omega}{\partial x} \mathrm{d}\Omega = \int_\Omega S\omega \mathrm{d}\Omega. \tag{3.7}$$

The highest order of derivative term present in the weak formulation (3.7) is the first order derivative term. Therefore the solution of the problem has to be one time differentiable in the weak sense i.e, it should be continuous. In addition it should satisfy the boundary conditions (3.3). The simplest function or the lowest order function that could satisfy (3.7) and (3.3) is a continuous piecewise function. Such function can have "kinks", e.g, on material interface but do not have any "jumps".

3.3 Finite element approximation

The next step is to divide the spatial domain, Ω into *n* subdomains i.e. Ω_i, $i = 1, 2\ldots, n$ (elements) with total $n+1$ nodes at locations x_i, $i = 0, 1\ldots, n$ (see Figure 8) such that the elements have lengths $x_i - x_{i-1}$. The element-wise approximation functions are the polynomial functions of order at least the order of the derivative term present in the weak form. We approximate the temperature distribution in elements using a first order polynom $T_i(x)$, $i = 1, 2\ldots, n$.

$$T_i(x) = \alpha_{i,1} + \alpha_{i,2}x, \qquad x_{i-1} \le x \le x_i. \tag{3.8}$$

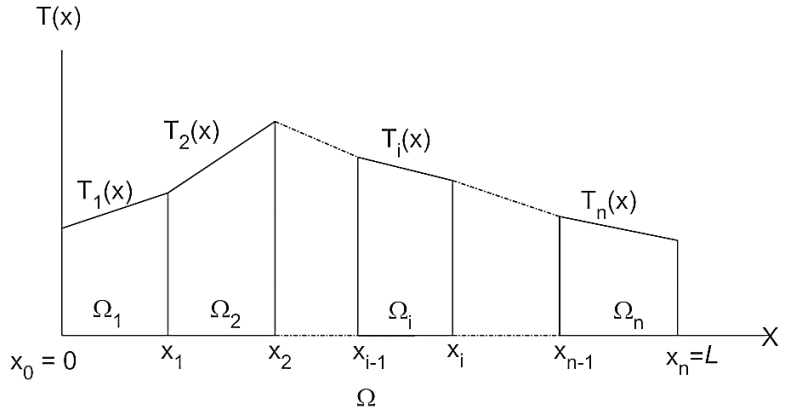


Figure 8 Schematical view of domain discretization and linear solution approximation.

Using so called basis functions (3.8) we evaluate the temperature at the nodes of the element *i*. We consider $k = 1, 2$ the local nodes of the element,

$$T_{i1} = \alpha_{i,1} + \alpha_{i,2}x_{i-1}, \tag{3.9}$$

$$T_{i2} = \alpha_{i,1} + \alpha_{i,2}x_i. \tag{3.10}$$

Rearranging (3.9) and (3.10) we get the constant terms,

$$\alpha_{i,1} = (x_i T_{i1} - x_{i-1}T_{i2})/(x_i - x_{i-1}),$$

$$\alpha_{i,2} = (-T_{i1} + T_{i2})/(x_i - x_{i-1}). \tag{3.11}$$

Solving (3.11) with (3.8), we get

$$T_i(x) = \frac{x_i - x}{x_i - x_{i-1}}T_{i1} + \frac{x - x_{i-1}}{x_i - x_{i-1}}T_{i2},$$

$$= \sum_{k=1}^{2} N_{ik}(x)T_{ik}, \tag{3.12}$$

where $N_{ik}(x)$, $k = 1, 2$, is the local element shape functions and is expressed as

$$N_{ik}(x) = \frac{(x - x_{i+1-k})(-1)^k}{x_i - x_{i-1}}. \tag{3.13}$$

Figure 9 shows the shape functions, $N_{ik}(x)$ plotted using (3.13).

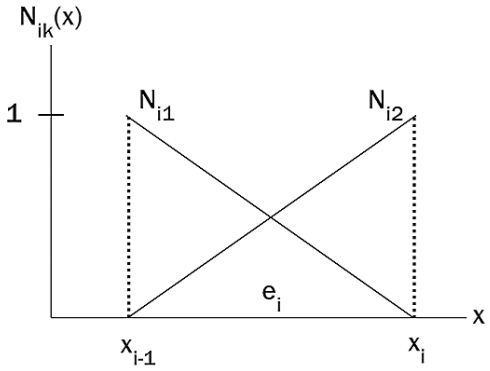


Figure 9 Linear shape functions N_{ik}, the local nodes of the element i, $k = 1, 2$ (3.13).

The nodal shape function can be plotted in a global coordinate system based on the local element shape functions. Figure 10 presents the nodal shape function $\varphi_i(x)$ at location x_i.

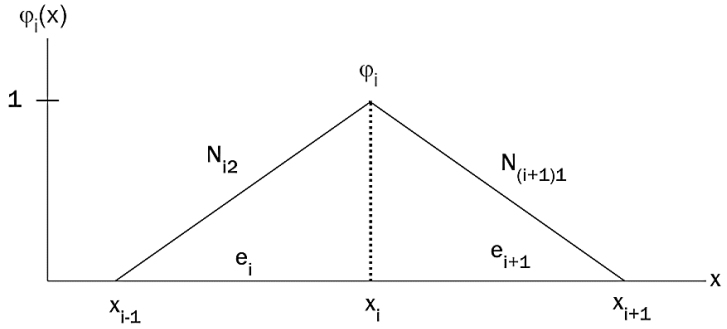


Figure 10 Nodal shape function φ_i (x) at location x_i.

From Figure 10 we see

$$\varphi_i(x) = \begin{cases} 0 \\ N_{i2} \\ N_{(i+1)1} \\ 0 \end{cases} = \begin{cases} 0 & \text{for } 0 \le x \le x_{i-1} \\ \frac{x - x_{i-1}}{x_i - x_{i-1}} & \text{for } x_{i-1} \le x \le x_i \\ \frac{x_{i+1} - x}{x_{i+1} - x_i} & \text{for } x_i \le x \le x_{i+1} \\ 0 & \text{for } x_{i+1} \le x \le L \end{cases}, \qquad i = 0, 1 \ldots, n. \tag{3.14}$$

Suppose the temperature degrees of freedom at node x_i and time t is $T_i(t)$, the finite element approximation of the temperature distribution throughout the spatial domain reads

$$T_h(x,t) = \sum_{i=0}^{n} \varphi_i(x)\, T_i(t). \tag{3.15}$$

3.4 Galerkin method to solve unknown degrees of freedom.

The unknown degrees of freedom $T_i(t)$ in the function (3.15) are solved by introducing the very approximating function as the weighting function in the weak form. We choose the $n + 1$ (number of nodes) weighting functions φ_j, $j = 0, 1\ldots, n$ to be the nodal shape functions (3.15) (the method known as Galerkin method) and rewrite the weak form (3.7),

$$\sum_{i=0}^{n}\left(\int_\Omega \rho c_\mathrm{p}\, \varphi_i \varphi_j \mathrm{d}\Omega\right)\frac{\mathrm{d}T_i}{\mathrm{d}t} + \sum_{i=0}^{n}\left(\int_\Omega k \frac{\mathrm{d}\varphi_i}{\mathrm{d}x}\frac{\mathrm{d}\varphi_j}{\mathrm{d}x}\mathrm{d}\Omega\right) = \int_\Omega S\varphi_j \mathrm{d}\Omega, \qquad \forall j. \tag{3.16}$$

The capacity matrix known as mass matrix and the thermal conductance matrix known as stiffness matrix are computed for $i, j = 0, 1, 2\ldots, n$ respectively as

$$\mathbf{C} = \left[\mathrm{C}_{ij}\right],\ \mathrm{C}_{ij} = \int_\Omega \rho c_\mathrm{p}\, \varphi_i \varphi_j \mathrm{d}\Omega,$$

$$\mathbf{K} = \left[\mathrm{K}_{ij}\right],\ \mathrm{K}_{ij} = \int_\Omega k \frac{\mathrm{d}\varphi_i}{\mathrm{d}x}\frac{\mathrm{d}\varphi_j}{\mathrm{d}x}\mathrm{d}\Omega. \tag{3.17}$$

Similarly the heat generation load known as force vector is computed for $i = 0, 1, 2 \ldots, n$ as

$$\mathbf{f} = \left[\mathrm{F}_j\right],\ \mathrm{F}_j = \int_\Omega S\, \varphi_j \mathrm{d}\Omega. \tag{3.18}$$

The unknown scalar values known as displacement vector $\mathbf{T} = [T_i]$ for $i = 0, 1, 2 \ldots, n$ can now be computed solving the simple equation system,

$$\mathbf{C}\dot{\mathbf{T}} + \mathbf{K}\mathbf{T} = \mathbf{f}, \tag{3.19}$$

with initial values $\mathbf{T_0} = [T(x_i, 0)] = \mathrm{g}(x_i)$, $i = 0, 1, 2\ldots., n$.

3.5 Time Integration

For transient problems the time domain ψ is considered to be infinite and we seek to find the solution for a certain interval e.g. $[0, t]$. The interval is discretized into m sub-intervals with time steps t_i, $i = 0, 1\ldots, m$ corresponding to unknown scalar values of temperature degrees of freedom, $\mathbf{T}_i$, $i = 0, 1\ldots, m$ such that the time step size is $\Delta t_i = t_i - t_{i\text{-}1}$, $i = 1, 2\ldots, m$. Assuming the previous values of temperature degrees of freedom, $\mathbf{T}_{i\text{-}1}$ are known, $\mathbf{T}_i$ is an unknown vector.

The approximation of the unknown temperature degrees of freedom and of its time derivative is given respectively as

$$\mathbf{T}_i(t) = \mathbf{T}_{i-1} + \frac{t - t_{i-1}}{\Delta t_i}(\mathbf{T}_i - \mathbf{T}_{i-1}),$$

$$= \mathbf{T}_{i-1} + \theta(\mathbf{T}_i - \mathbf{T}_{i-1}), \qquad i = 1, 2\ldots, m, \qquad (3.20)$$

and

$$\dot{\mathbf{T}}_i(t) = \frac{\mathbf{T}_i - \mathbf{T}_{i-1}}{\Delta t_i}, \qquad i = 1, 2\ldots, m, \qquad (3.21)$$

where $\theta = (t - t_{i-1}) / \Delta t_i$ is known as weighting parameter. Updating (3.19) with (3.20) and (3.21) we get

$$\frac{1}{\Delta t_i}\mathbf{C}(\mathbf{T}_i - \mathbf{T}_{i-1}) + \mathbf{K}[\mathbf{T}_{i-1} + \theta(\mathbf{T}_i - \mathbf{T}_{i-1})] + \bar{\mathbf{f}} = 0, \qquad (3.22)$$

where $\bar{\mathbf{f}} = \mathbf{f}_i + \theta(\mathbf{f}_i - \mathbf{f}_{i-1})$ is assumed to vary linearly over the time step. Rearranging (3.22) gives the expression of the unknown vector [26],

$$\mathbf{T}_i = (\mathbf{C} + \theta\Delta t_i\mathbf{K})^{-1}[(\mathbf{C} - (1-\theta)\Delta t_i\mathbf{K})\mathbf{T}_{i-1} - \Delta t_i\bar{\mathbf{f}}]. \qquad (3.23)$$

Different values of θ give different schemes of time stepping: θ = 0 (forward difference or *explicit* Euler); θ = ½ (*semi-implicit* or Crank Nicholson); θ = 1 (backward difference or *implicit* Euler) [28-30]. The commercial finite element packages usually use Newmark method and General Hilber-Hughes-Taylor method for transient analysis, detail information can be found in [31]. Adaptive time stepping methods are incorporated in some popular software packages e.g. ANSYS, COMSOL. The quench simulation of large magnets needs adaptive methods in order to reduce the computational cost.

3.6 Non-Linearity

In our case the specific heat capacity, thermal conductivity and the heat generation load have non-linear dependency on temperature for which the system equation (3.19) takes the form,

$$\mathbf{C}(\mathbf{T})\dot{\mathbf{T}} + \mathbf{K}(\mathbf{T})\mathbf{T} = \mathbf{f}(\mathbf{T}). \qquad (3.24)$$

As can be seen from (3.24) the material properties change with the applied load. To solve such problems an iterative method, the Newton-Raphson is used [32-34]. The method goes through several iterations within a time step and updates the material matrices. The process is explained in the following paragraph.

Provided the initial value of temperature degrees of freedom $\mathbf{T}_{i-1}$ is given, the corresponding capacity matrix $\mathbf{C}_{i-1}$ and the stiffness matrix $\mathbf{K}_{i-1}$ will be always known. If the material properties are assumed to vary linearly or remain constant over a time step, the equilibrium

equation (3.24) is not fulfilled throughout the time step. The amount by which the equilibrium is out of balance is called residual vector given by

$$\mathbf{R}_j(\mathbf{T}) = \mathbf{f}_j - \mathbf{C}_{i-1,j}\dot{\mathbf{T}} + \mathbf{K}_{i-1,j}\mathbf{T}. \tag{3.25}$$

Note: i represents time steps and j = 1, 2…, l represents iteration steps. The approximate solution to (3.24) is obtained by repeating

i) Compute $\mathbf{C}_{i\text{-}1,\,j}$, $\mathbf{K}_{i\text{-}1,\,j}$ and $\mathbf{f}_{\,j}$ for $\mathbf{T}_j$; for first iteration $\mathbf{T}_j = \mathbf{T}_1 = \mathbf{T}_{i\text{-}1}$,

ii) Improve the solution using Newton formula

$$\mathbf{T}_{j+1} = \mathbf{T}_j - \left[\frac{\mathbf{dR}_j(\mathbf{T})}{\mathbf{dT}}\right]^{-\mathbf{1}} \mathbf{R}_j(\mathbf{T}), \tag{3.26}$$

until the absolute or relative L_2(Euclidean) norm of the residual vector is below the specified tolerance ϵ,

$$\left\|\mathbf{R}_j\right\|_2 \leq \epsilon, \text{ or } \frac{\left\|\mathbf{T}_{j+1}-\mathbf{T}_j\right\|_2}{\left\|\mathbf{T}_{j+1}\right\|_2} \leq \epsilon. \tag{3.27}$$

4. SUPERCONDUCTING STRAND 1D MODEL

Superconducting strand is the basic element of the magnet coil. The simulation of a quench in the strand helps to understand the overall picture of quench behavior and possible damage that it can cause in the magnet. In order to detect a quench, the resistive voltage across the magnet is measured and compared to a so called quench threshold; see Appendix 2. The results obtained from the 1D model of the superconducting wire can be used to calculate the quench thresholds. Increased accuracy of the model helps adjusting parameters of the quench protection system which in turn contributes in the optimization of the magnet operation. Therefore so much emphasis is given to build and improve the quench simulation model of the superconducting wire.

In this chapter we explain the numerical model of the superconducting wire constructed in commercial software packages. For this we choose two popular simulation environments i.e. ANSYS and COMSOL. We take QP3 as reference to compare and validate our models. QP3 is the FORTRAN code developed at CERN. The model represents superconducting wire as a longitudinally discretized 1D network model. The code estimates the temperature profile of the superconducting wire in the time domain. For more information see [8].

The numerical model simulates the quench behavior on the strand of the inner coil of the LHC dipole magnets. The strand diameter is 1.065 ± 0.0025 mm. The strand has 8800 ± 20 number of Nb-Ti filaments of diameter 7 ± 1 μm. The copper to superconductor volume ratio is 1.65 ± 0.03 [1].

The Nb-Ti filaments do not contribute to the electrical conduction after a quench, so heat is produced only in the copper cross-section. On the other hand, since the temperature is assumed to be homogeneous across the wire, the Nb-Ti fraction contributes to the overall heat capacity. One could therefore build a model with actual diameter of the strand and change the formula for heat production and heat conductivity or assume the model to contain only copper fraction and change the formula for the heat capacity. We choose latter option. The diameter of the assumed cylindrical strand domain, d_{Cu} is therefore smaller than the actual diameter of the wire, d_{wire} and is calculated as

$$d_{\mathrm{Cu}} = \sqrt{\frac{f}{1+f}}\, d_{\mathrm{wire}}, \qquad [\mathrm{m}] \qquad (4.1)$$

where f is the copper to superconductor volume ratio.

LHC dipole magnets are made up of Rutherford type cable. Figure 3d presents the arrangement of the strands in the cable which suggests that the cylindrical surface of the strand is not fully surrounded by liquid helium. We assume V_{he} volume of helium surrounds the conductor and the total contact surface area is $a_{He} = s * L$, where L is the length of the conductor and s is the arc length parameter to assign an appropriate contact surface area between the conductor and the helium. The cross-sectional properties corresponding to a particular strand of the inner layer of the LHC dipole magnet's coils and the values of the parameters used to model the presence of helium around the strand is summarized in Table 1. The V_{he} is chosen for academic purpose, to enhance the effect of helium cooling. It does not reflect the actual amount of helium present around the wire.

Table 1 Geometry parameters used for the quench simulation of 1D adiabatic model.

Parameter	Symbol	Value	Unit
Length of the strand	L	1	m
Diameter of the strand	d_{wire}	1.065e-3	m
Copper to superconductor ratio	f	1.65	-
Total volume of helium	V_{He}	1e-6	m^3
Contact surface area	s	1.320e-3	m

4.1 Material Properties

The thermal conductivity of the superconductor material is negligible as compared to that of copper; hence is ignored [10]. The thermal conductivity of copper, k_{Cu} has a non-linear dependency on the magnetic field and temperature; see Appendix 9. The heat capacity of the copper, $c_{p,Cu}$, and superconductor, $c_{p,NbTi}$, is temperature-dependent; see Appendix 7 and 8. The equivalent heat capacity of the strand is calculated as

$$c_{\mathrm{p,strand}}(T) = f_{\mathrm{Cu}} c_{\mathrm{p,Cu}}(T) + (1 - f_{\mathrm{Cu}}) c_{\mathrm{p,NbTi}}(T), \qquad [\mathrm{Jm^{-3}}] \qquad (4.2)$$

where $f_{Cu} = f/(1+f)$ is the fraction of copper volume in the strand. Since the geometry of our model compromises only the copper volume, the heat capacity to be defined to the model is obtained as

$$c_{\mathrm{p,model}}(T) = \frac{c_{\mathrm{p,strand}}(T)}{f_{\mathrm{Cu}}} = c_{\mathrm{p,Cu}}(T) + \frac{1 - f_{\mathrm{Cu}}}{f_{\mathrm{Cu}}} (c_{\mathrm{p,NbTi}}(T)),$$

$$= c_{\mathrm{p,Cu}}(T) + \frac{1}{f} c_{\mathrm{p,NbTi}}(T). \qquad [\mathrm{Jm^{-3}}] \qquad (4.3)$$

The thermal conductivity, density and the heat capacity of the helium are temperature-dependent; see Appendix 10, 11 and 12.

COMSOL has so-called “function tools” that allow to define material properties as a function of desired field variables. Similarly in ANSYS a table can be created with field variable values in one column and the corresponding material property values in another column. Such a function or a table is then assigned to the model such that the solver updates the material property matrices every iterations based on the nodal values of field variables.

4.2 Adiabatic Model

In Chapter 2 we presented five different boundary value problems involved in the quench. In this section we present a model constructed in COMSOL and in ANSYS to solve the one of the problems, i.e the thermal problem in the solid. The model do not take into account the presence of helium around the conductor and the heat diffusion is confined to the conductor domain only.

4.2.1 Geometry and discretization

The cylindrical conductor domain is discretised into n segments with $n + 1$ nodes as presented in Figure 11.

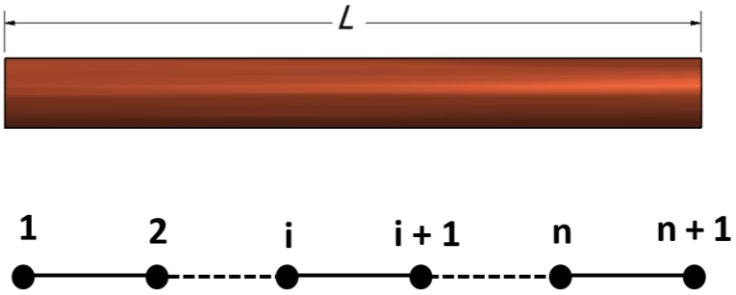


Figure 11 Assumed geometry of the superconducting wire.

In COMSOL, a straight line was created via the Graphical User Interface (GUI) and the “meshing tool” was used to discretise the domain. The cross-section properties were defined in the domain definition section. We specify the problem type as a transient thermal analysis for which the system equation to be calculated becomes equivalent to (3.24).

In ANSYS, dedicated APDL scripts were developed instead. Firstly, nodes were created along a coordinate axis. Then, the nodes were connected with appropriate elements. We chose the two-noded uniaxial thermal element FLUID116. It conducts heat between its nodes. The element has two different types of degrees of freedom: temperature and pressure. We use only the temperature degrees of freedom, for which the equilibrium equation of the elements

becomes equivalent to (3.24). ANSYS uses linear approximation function for this element; such function is explained in Section 3.3.

The two main reasons why this element was selected are:

i) Material tables can be directly assigned to the element; see Section 4.1.

ii) It is a two noded element with possibility to apply surface loads.

4.2.2 Initial condition

We specify temperature at each node as an initial condition such that the temperature profile is of Gaussian shape with a peak value at x = 0 m. Such a temperature distribution mimics the initial heat pulse. It is calculated as

$$T(x) = \mathrm{T_{op}} + (\mathrm{T_{peak}} - \mathrm{T_{op}})e^{\left(-\left(\frac{x}{\alpha}\right)^2\right)}, \qquad [\mathrm{K}] \qquad (4.4)$$

where $\mathrm{T_{op}}$ is the operating temperature and $\mathrm{T_{peak}}$ is the hot-spot temperature, x is the node position and α is the constant parameter that allows to adjust the shape of the Gaussian curve. The curves in Figure 12 represent the initial temperature profile for two different values of α.

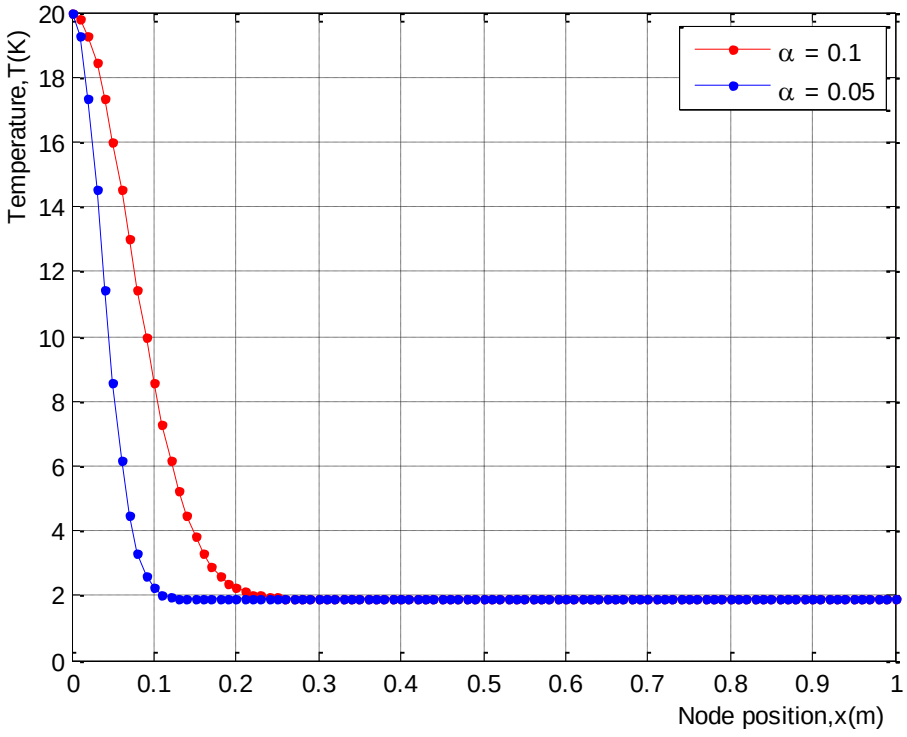


Figure 12 Gaussian type temperature profile representing the initial heat pulse.

The hot-spot temperature specified is well above the current-sharing temperature in order to provoke a quench.

4.2.3 Loads and Boundaries

For the adiabatic model the load is Joule heating given by expression (2.8). Our model is a thermal model and considers only temperature as a field variable. The magnetic field and current are introduced as constant values in order to reproduce conditions prior quench detection. Hence, the equation of Joule heat generation is a function of one field variable, i.e, temperature, T which is implemented as a table or a function.

The boundary condition in our case is zero heat flux on both ends. For 1D transient thermal analysis it is the default setting in both COMSOL and ANSYS.

4.2.4 Solution and Results

The transient thermal analysis was carried out from 0 to t s. An automatic time stepping option with minimum time step of 10 μs and maximum time step of 100 μs was specified to the solver. Such scheme uses adaptive time stepping with time step size between specified minimum and maximum value. To deal with material non-linearity the Newton-Raphson iteration scheme was selected.

The current and magnetic field values selected corresponds to the strand of the inner coil of the dipole magnet. Its location is indicated by black dot in the Figure 13. The strand being close to beam is likely to quench due to beam induced losses.

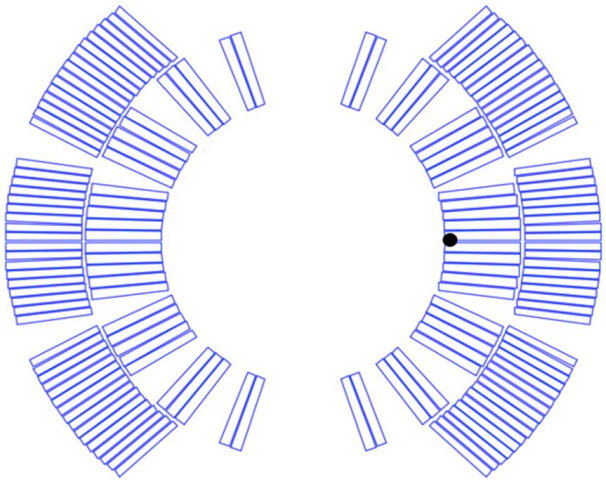


Figure 13 Cross-section of the coil of LHC dipole magnets. Black dot indicate the location of the strand whose cross-section properties, nominal current and magnetic field distribution are taken as a reference for the simulation.

Figure 14 represents the results obtained from COMSOL for the setting summarized in Table 2. The value for T_{peak} and α are arbitrary chosen and may not represent the actual

temperature distribution corresponding to the heat generated due to the induced losses or frictional movement.

Table 2 Additional parameters used for the simulation of the 1D adiabatic model.

Parameter	Symbol	Value	Unit
Discretization	n	100	-
Transport current	I	150	A
Magnetic field	B	2.88	T
Operating temperature	T_{op}	1.9	K
Hotspot temperature	T_{peak}	20	K
Gaussian parameter	α	1.5	m

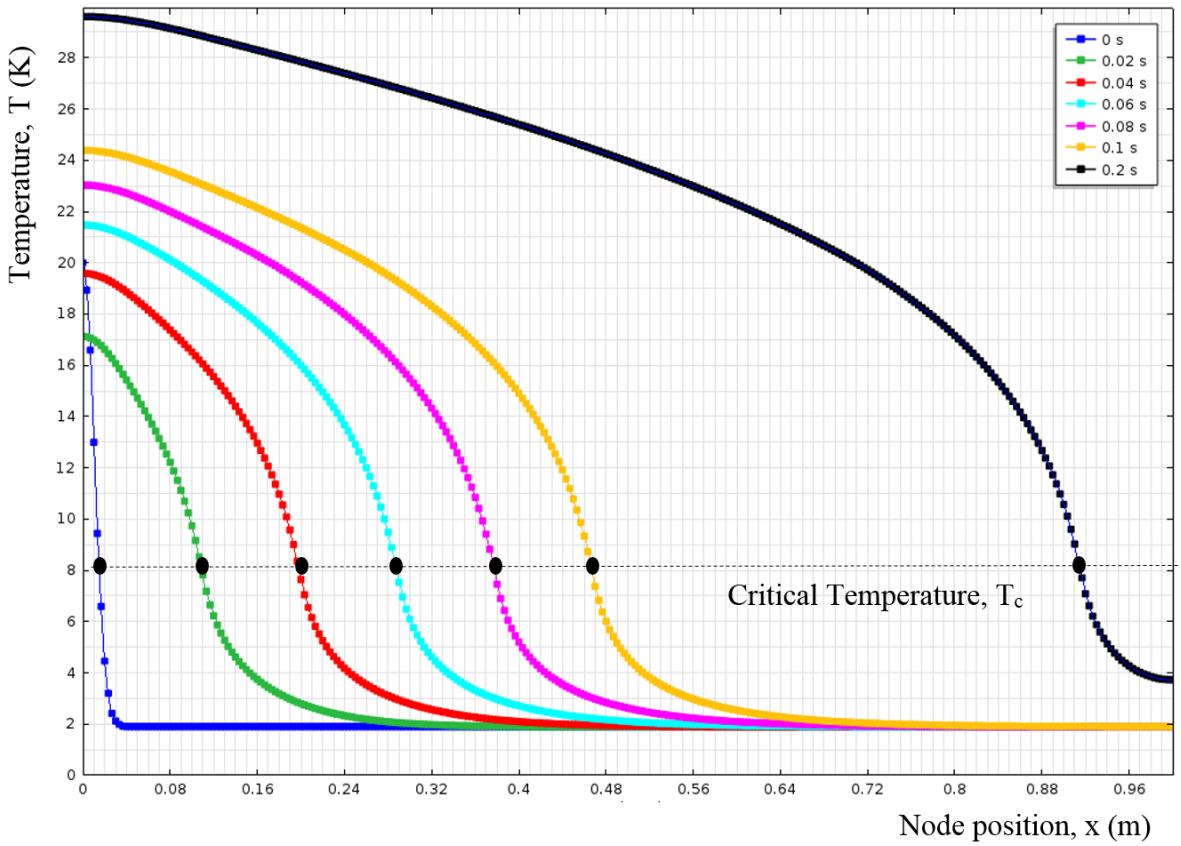


Figure 14 Temperature distribution in the superconducting wire plotted for different time. The black dots depicts advancing quench front.

Since the specified peak temperature is greater than the critical temperature, the quench always initiates at the hot-spot. The quench may propagate further or vanish depending upon the intensity of the heat pulse and the Joule heating corresponding to the transport current. On the one hand the heat conduction phenomenon in the wire decreases the hot-spot temperature, on the other hand the Joule effect increases the temperature of a node that is in the normal conducting state. Since usually at the onset of a quench only a small fraction of strand is above

the critical temperature, in the beginning of the transient, the heat conduction phenomenon becomes more effective compared to Joule heating and the hot-spot temperature drops. If the intensity of heat pulse i.e. area covered by Gaussian curve is below a certain limit, the initial drop in hot-spot temperature can fall below the critical temperature thereby preventing quench propagation.

In our case one can notice from the figure that initially the hot-spot temperature drops however does not fall below the critical temperature thus leading to quench zone expansion and further heat generation. The generated heat propagates longitudinally successively raising the temperature above critical temperature and propagating the quench.

The dotted horizontal line shown in Figure 14 represents calculated critical temperature which is 8.073 K. We can calculate the quench propagation speed by plotting the position of the quench front as a function of time i.e. the position of black dots in Figure 14 plotted against the corresponding time; see Figure 15.

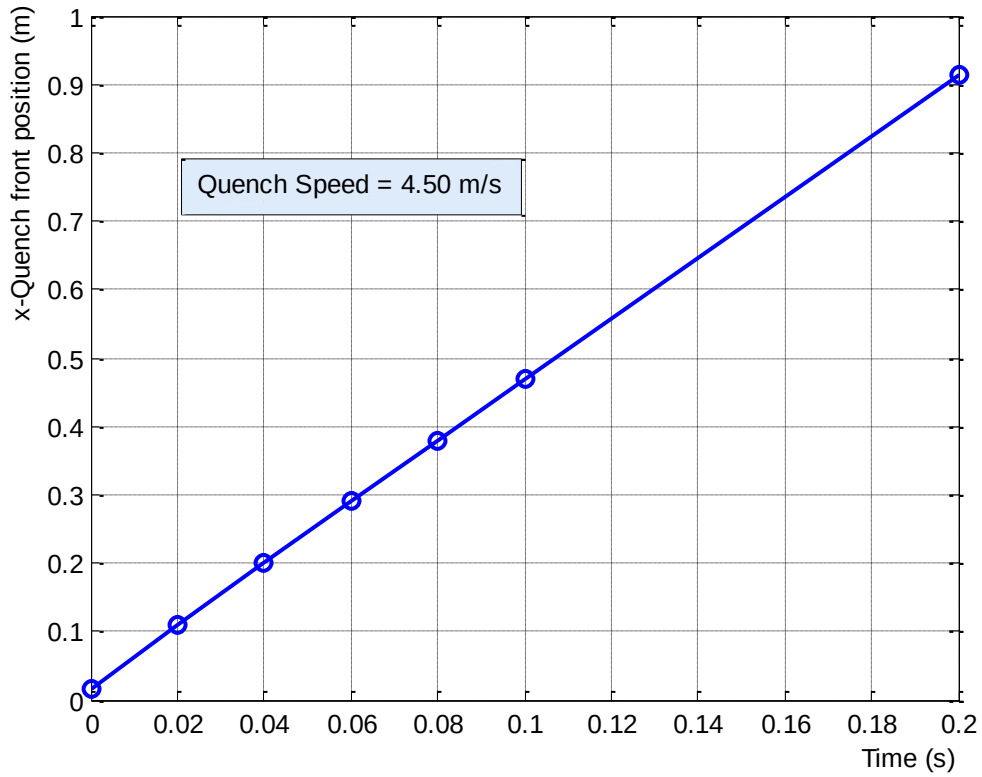


Figure 15 Quench front position as a function of time in reference to Figure 14.

The curves in Figure 16 present the results obtained from different simulation environments at time t = 0.2 s for the settings summarized in Tables 1 and 2.

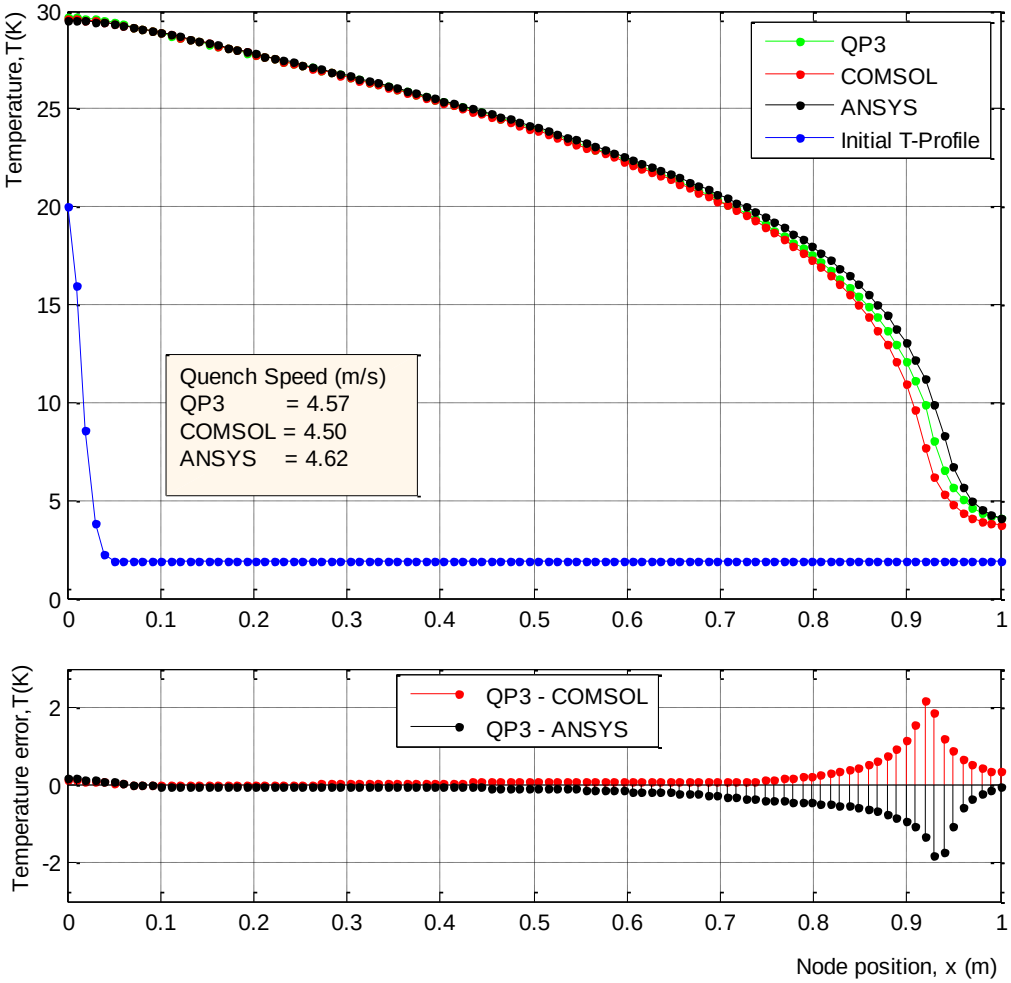


Figure 16 Upper: Temperature profile of the superconducting wire estimated by adiabatic models of different simulation environment, Lower: Difference in Temperatures profile in reference to QP3.

The red and black curves in the sub-plot is the calculated temperature difference between ANSYS and QP3 and COMSOL and QP3 respectively. For the quenching strand it is likely to have significant temperature difference at quench front position where there is rapid temperature rise because of quench zone expansion. One can notice the phenomena from Figure 16 at x = ~ 0.93 m. We are mainly interested in correctly estimating the hot-spot temperature and quench propagation speed for which satisfactory results were obtained. Table 3 shows the difference in hot-spot temperature and quench propagation speed in reference to QP3.

Table 3 Variation in hot-spot temperature and quench speed in reference to QP3. Results corresponds to Figure 16.

Simulation Environment	Hot-spot temperature difference (K)	Quench propagation speed difference (m/s)
ANSYS	0.18 (0.60%)	0.05 (1.09%)
COMSOL	0.11 (0.37%)	0.07 (1.53%)

To find out the appropriate scale of spatial discretization we did a convergence analysis for the same settings summarised in Tables 1 and 2. The upper plot in Figure 17 presents the

quench propagation speed calculated for different mesh refinements, i.e, *n* values. The quench propagation speed was calculated based on the initial and final positions of the quench front. The minimum and maximum time stepping specified was 10 µs and 500 µs respectively.

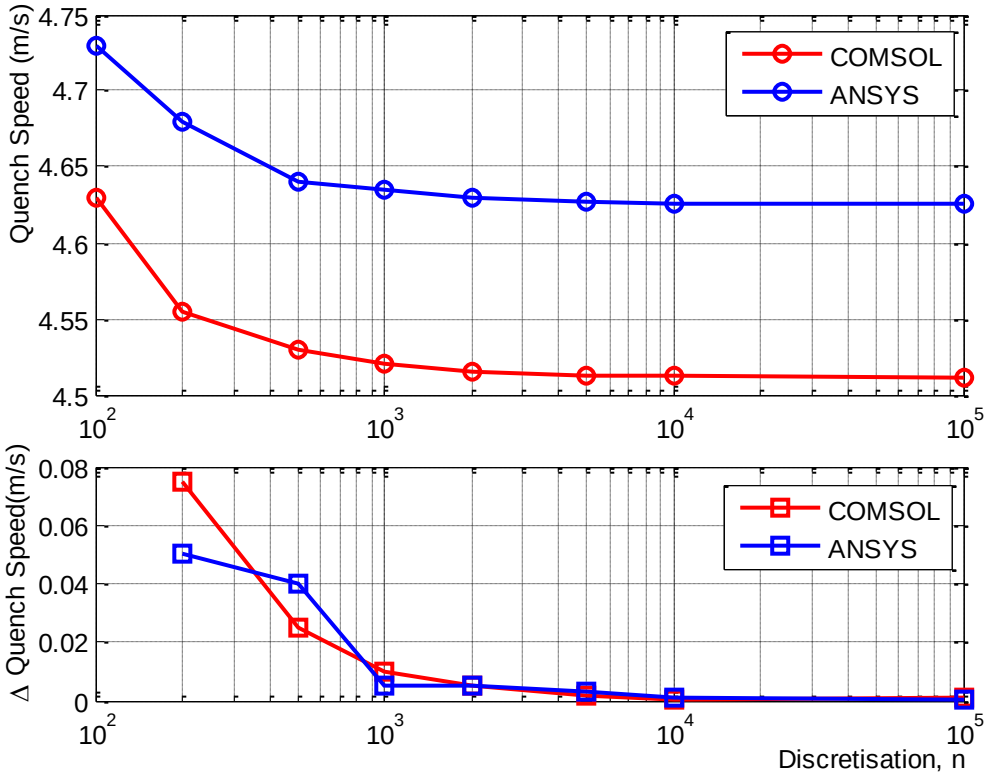


Figure 17 Upper: Quench speed plotted for different mesh refinements, Lower: Change in quench speed with respect to mesh refinement.

The lower plot calculates the difference in quench propagation speed between two adjacent *n* values. The converged solution between two simulation environments varies by 0.114 m/s (2.49 %). At $n = 1000$ we reached the desired accuracy, i.e, ≤ 1 % change in quench speed in both simulation environments. Therefore discretization in the scale less than or equal to a millimetre is recommended for the adiabatic scenario of quench simulation.

Figure 18 compares the performance of different simulation environments in terms of calculation time. The total computation time was plotted for different mesh refinements. The Curves correspond to the convergence plot shown in Figure 17. The dashed lines indicates the linear behaviour of the curves. The analysis were performed in an eight core 3.7 GHz PC computer with 32 GB RAM (Random Access Memory).

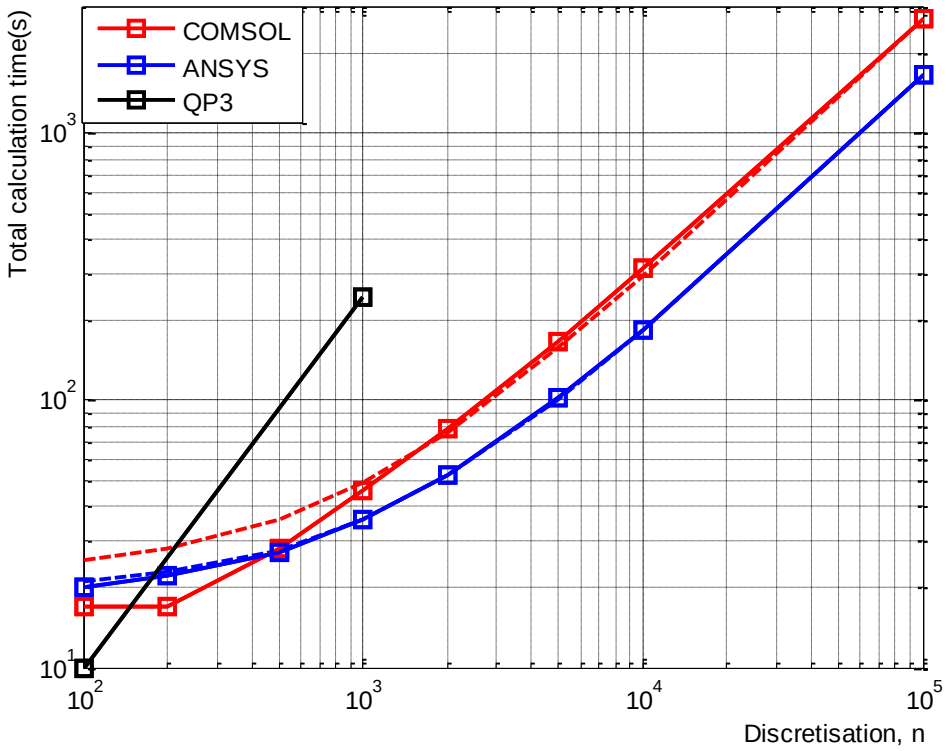


Figure 18 Total computation time plotted for different mesh refinements. The plotted results corresponds to the convergence plot presented in Figure 17. Dashed lines indicates the linear behaviour of the curves.

One can notice from Figure 18 that for large number of elements ANSYS starts to perform better. Since we need to discretise the strand in the scale less than or equal to a millimetre and have to simulate much more than one meter of wire there will be more than 1000 elements in the model for which the difference between COMSOL and ANSYS is of about 10%.

4.3 Helium-cooled model

In this section we present the models constructed in ANSYS in order to solve the two of the five boundary value problems, i.e, A) the thermal problem in the conductor B) heat transfer between the conductor and helium. The models solve the problems simultaneously as a coupled physical problems.

The thermal problem in the conductor, A is already solved, i.e, the adiabatic model; see Section 4.2. The heat transfer between the conductor domain (numerically the adiabatic model) and the helium domain can be modeled with contact elements (representing convection, radiation and/or conduction) on the interface; see Figure 19. ANSYS allows the heat transfer coefficient (HTC) of such contact elements to be a function of one among following three variables:

i) Temperature of one of the domain.

ii) Average temperature of both domains.

iii) Temperature difference between the domains.

With the above options it is not possible to implement the HTC that is equivalent to empirical equation representing the heat transfer between conductor and helium (2.14). This shows that the standard library of ANSYS is not generic enough to implement the complicated heat transfer equation. Nevertheless, one can create a custom algorithm within ANSYS simulation platform, using APDL scripts, and force ANSYS standard routine (transient thermal analysis) to run several times within a loop such that every time steps appropriate heat transfer parameters, e.g, HTCs are calculated and applied to the corresponding elements. In the following sections we present three different approaches to update the heat transfer parameters. We distinguish them as three different models.

4.3.1 Model 1: Switching of contact elements

The superconducting wire and the liquid helium are modelled as two parallel cylindrical domains each discretised into n segments with $n + 1$ nodes as shown in Figure 19. The dimension of the conductor domain is the same as in the adiabatic model. We use the thermal element FLUID116 in both domains in order to calculate the heat conduction along the longitudinal direction (3.24).

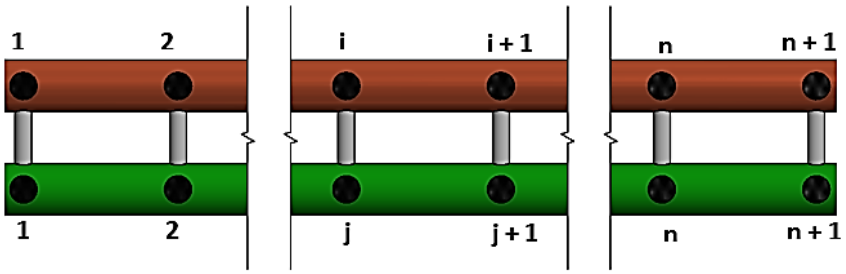


Figure 19 Model 1 helium cooled model.

By assigning material properties; heat capacity, density, thermal conductivity and the viscosity to the helium domain it is possible to model the heat transfer phenomena within the helium channel (bulk helium) which is in contrast with the heat transfer phenomena between the conductor and the helium. The laminar (Landau) regime and the turbulent (Gorter-Mellink) regime explains the heat transfer through the helium channel [19]; See Appendix 3. QP3 however does not consider heat transfer within the helium domain. To have a good agreement in the result compared to QP3 the corresponding material properties of helium domain were switched off.

The heat transfer between the domains is explained by 5 different regimes; see Section 2.1 Two different types of ANSYS contact elements were used, i.e, LINK31 to represent the Kapitza regime and LINK34 to represent the remaining regimes.

I. Radiation Link - LINK31

LINK31 is a uniaxial two noded thermal element. It models the radiation heat flow rate between its nodes. The equation is of the form

$$h = \sigma\epsilon\left(\mathrm{F}T_i^4 - \mathrm{A}T_j^4\right), \qquad [\mathrm{Wm}^{-2}] \tag{4.5}$$

where σ is the Stephan-Boltzmann constant, ϵ is the emissivity and F and A are additional material constants. We assume $\sigma\epsilon = a_{kap}$, $F = T_i^{nkap-4}$ and $A = T_j^{nkap-4}$ such that (4.5) becomes equivalent to heat flow equation of the Kapitza regime (2.9).

II. Convection Link - LINK34

LINK34 is also a uniaxial two noded thermal element. It convects heat between its nodes. The convection heat flow rate for this element reads

$$h = h_{\mathrm{f}}(T_i - T_j)^{\mathrm{n_e}}, \qquad [\mathrm{Wm}^{-2}] \tag{4.6}$$

where h_f is the HTC and n_e is the empirical constant. Table 4 presents the assumed material constants for the different regimes such that the equation (4.6) becomes equivalent to the corresponding the heat flow equation.

Table 4 Contact elements and their corresponding real constants.

Regime	HTC, $h_f(Wm^{-2}K^{-n_e})$	Empirical constant, n_e
Film-Boiling II	a_{FB-II}	1
Natural Convection	a_{NC}	1
Nucleate Boiling	a_{NB}	2.5
Film-Boiling I	a_{FB-I}	1

The heat transfer surface area defined to the contact elements is equivalent to the total contact area between the conductor and helium. In this model we have three different types of elements, seven different types of materials and in total $2n + 5(n + 1)$ number of elements.

The imposed loads and the initial condition for the conductor domain, i.e, nodes $i = 1, 2\ldots, n$ are the same as defined in the adiabatic model. The initial temperature defined to the helium domain i.e. to nodes $j = 1, 2\ldots, n$ is equal to $T_{op.}$

Figure 20 shows the block diagram of the algorithm built to impose a governing regime of heat transfer during the analysis. The code can be found in the Appendix 16. The time steps are represented as $k = 1, 2\ldots, m$.

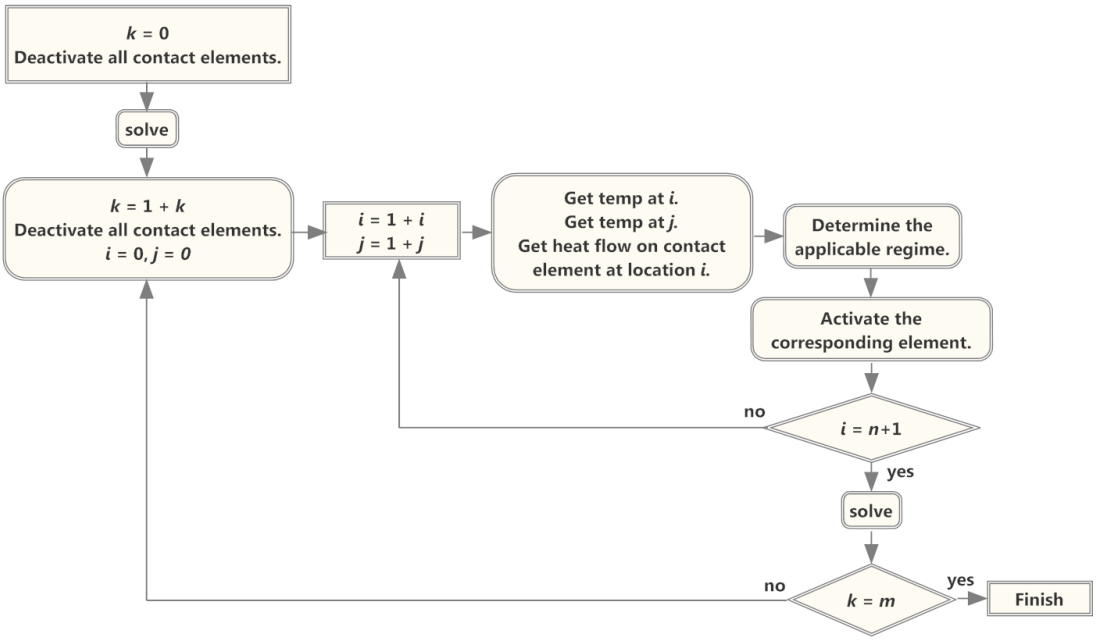


Figure 20 Block diagram of the algorithm that switches appropriate element every time steps.

To take into account all regimes of heat transfer we join 5 contact elements of different material constants between every pair of nodes such that each contact element represents a single regime. In the beginning of the analysis all contact elements are inactive. At every time step, the temperatures at the nodes and the heat flow rates in the contact elements is extracted from the solutions of the previous time step. Based on extracted temperature values and the heat flow rates an applicable regime of the heat transfer is determined for each node pair and the corresponding contact element is activated. The algorithm allows only one contact element to be active at the same time. For each node pair the analysis starts with the lowest regime, stays until the heat flux is above the limit, and shifts to another higher regime with a condition that it can fall back if the heat flux drops below the limit.

4.3.2 Model 2: Updating heat transfer coefficient

The cylindrical conductor domain is discretized as in the adiabatic model. Its nodes are linked to helium masses of volume $V_{he}/(n+1)$ as shown in Figure 21.

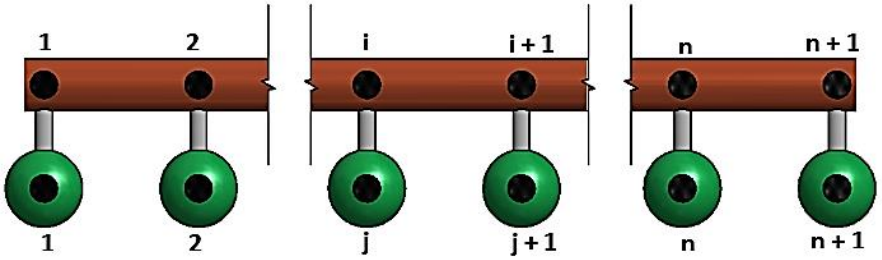


Figure 21 Model 2 helium cooled model.

The link is a contact element LINK34. One-noded mass elements MASS71 are used to model the helium masses. In this scheme the heat conduction along the helium is not taken into consideration. Unlike in the previous model only one contact element is connected between each node pairs. There are altogether $n + 2(n+1)$ elements in the model.

The load and the initial condition in the conductor domain and the initial condition in the helium masses are the same as in the previous model (Model 1). The heat flow rate in the contact element is of the form (4.6). We set the empirical constant $n_e = 1$ and redefine the flow rate as

$$h^{k+1} = {h_f}^{k+1}\left(T_i{}^k, T_j{}^k\right)\left(T_i{}^{k+1} - T_j{}^{k+1}\right), \qquad [\mathrm{Wm^{-2}}] \qquad (4.7)$$

where $k = 1, 2\ldots, m$ represents time steps. The (5.7) becomes equivalent to (2.14) with

$${h_f}^{k+1}\left(T_i{}^k, T_j{}^k\right) = \begin{cases} a_{kap}\dfrac{\left(T_i^{n_{kap},\,k} - T_j^{n_{kap},\,k}\right)}{T_i{}^k - T_j{}^k} & T_j{}^k < T_\lambda \;\&\; h^k < Q_{kap} \\ a_{FB\text{-}II} & T_j{}^k < T_\lambda \;\&\; h^k \geq Q_{kap} \\ a_{NC} & T_j{}^k \geq T_\lambda \;\&\; h^k < Q_{NC} \\ a_{NB}\left(T_i{}^k - T_j{}^k\right)^{1.5} & T_j{}^k > T_\lambda \;\&\; h^k > Q_{NC} \\ a_{FB\text{-}I}\left(T_i{}^k - T_j{}^k\right) & T_j{}^k > T_\lambda \;\&\; h^k \geq Q_{NB} \end{cases} \cdot \quad [\mathrm{Wm^{-2}}] \qquad (4.8)$$

The algorithm to update the HTC of the contact elements is depicted in Figure 22.

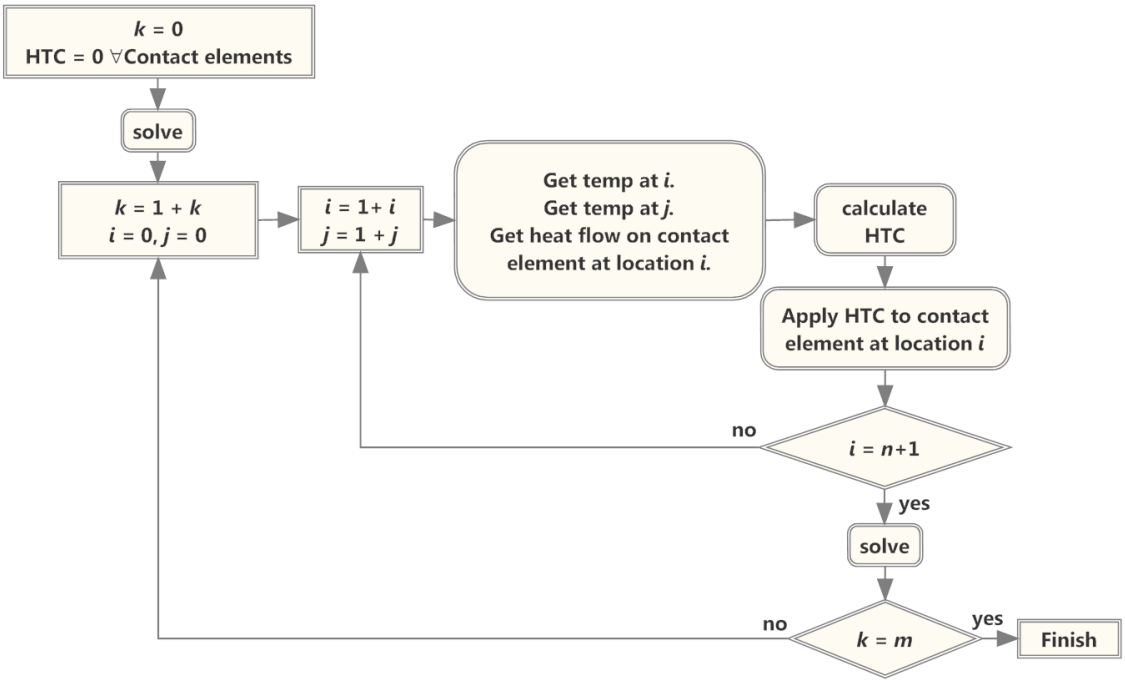


Figure 22 Block diagram of the algorithm that calculates heat transfer coefficient of the contact elements every time steps.

At every time step we extract the temperature at nodes and the heat flow rates in the contact elements. Based on the values obtained we calculate HTC for each contact elements using (4.8).

4.3.3 Model 3: Updating the load

The model is similar to the adiabatic model. We do not represent the helium as a separate domain, instead we introduce the helium cooling effect as negative heat generation load in the equilibrium equation. The amount of heat flow to the liquid helium in contact is subtracted from the Joule heat generated at nodes i = 1, 2…, n + 1. In Figure 23 the time steps are represented as k = 1, 2…, m and j = 1, 2…, n+1 is used to define the temperature of liquid helium in contact at locations corresponding to the nodes of the conductor domain.

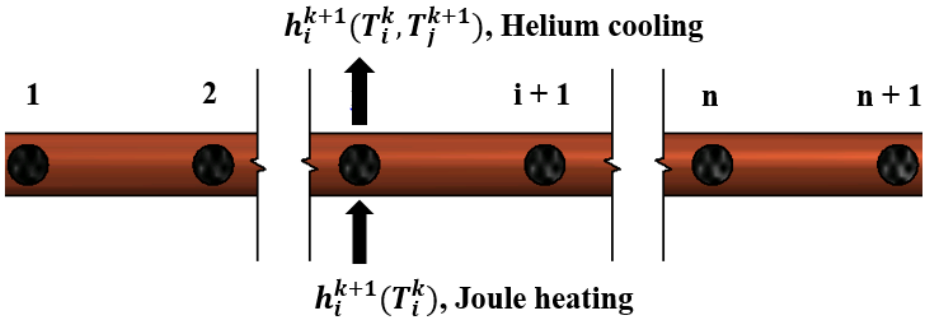


Figure 23 Model 3 Helium cooled model.

Given the initial temperature, material properties and the volume of the liquid helium, the temperature of the liquid helium in contact at several locations for different time steps is expressed as

$$T_j^{k+1} = \frac{h_{\mathrm{He}}^k a_{\mathrm{He}}}{\rho_{\mathrm{He}}(T_j^k) c_{\mathrm{p,He}}(T_j^k) V_{\mathrm{He}}} \Delta t + T_j^k, \qquad [\mathrm{K}] \qquad (4.9)$$

where ρ_{He} is the density, $c_{\mathrm{p,\,he}}$ the heat capacity and V_{he} the total volume of the liquid helium. h^k_{He} is the amount of heat flow to liquid helium, a_{He} the surface area of the conductor in contact with liquid helium and Δt the time step size.

The initial temperature of the nodes in the conductor domain is the same as in the adiabatic model and initial temperature of liquid helium in contact, $T_j^1 = \mathrm{T_{op}}$. The net heat generation load to be updated at every time step reads

$$S_i^{k+1}(T_i^k, T_j^{k+1}) = h_{\mathrm{Joule}}^{k+1}(T_i^k) - \frac{a_{\mathrm{He}}}{a_{\mathrm{Cu}} L} h_{\mathrm{He}}^{k+1}(T_i^k, T_j^{k+1}), \qquad [\mathrm{Wm^{-3}}] \qquad (4.10)$$

where a_{Cu} is the cylindrical surface area calculated considering only copper volume and L is the length of the conductor. The Joule heating function, $h_{\mathrm{Joule}}{}^{k+1}(T_i{}^k)$ for constant magnetic field can

be calculated using (2.8) and the rate of heat flow, $h_{He}^{k+1}(T_i^k, T_i^{k+1})$ can be calculated using (2.14). Figure 24 presents the block diagram of the overall process of updating the load vector.

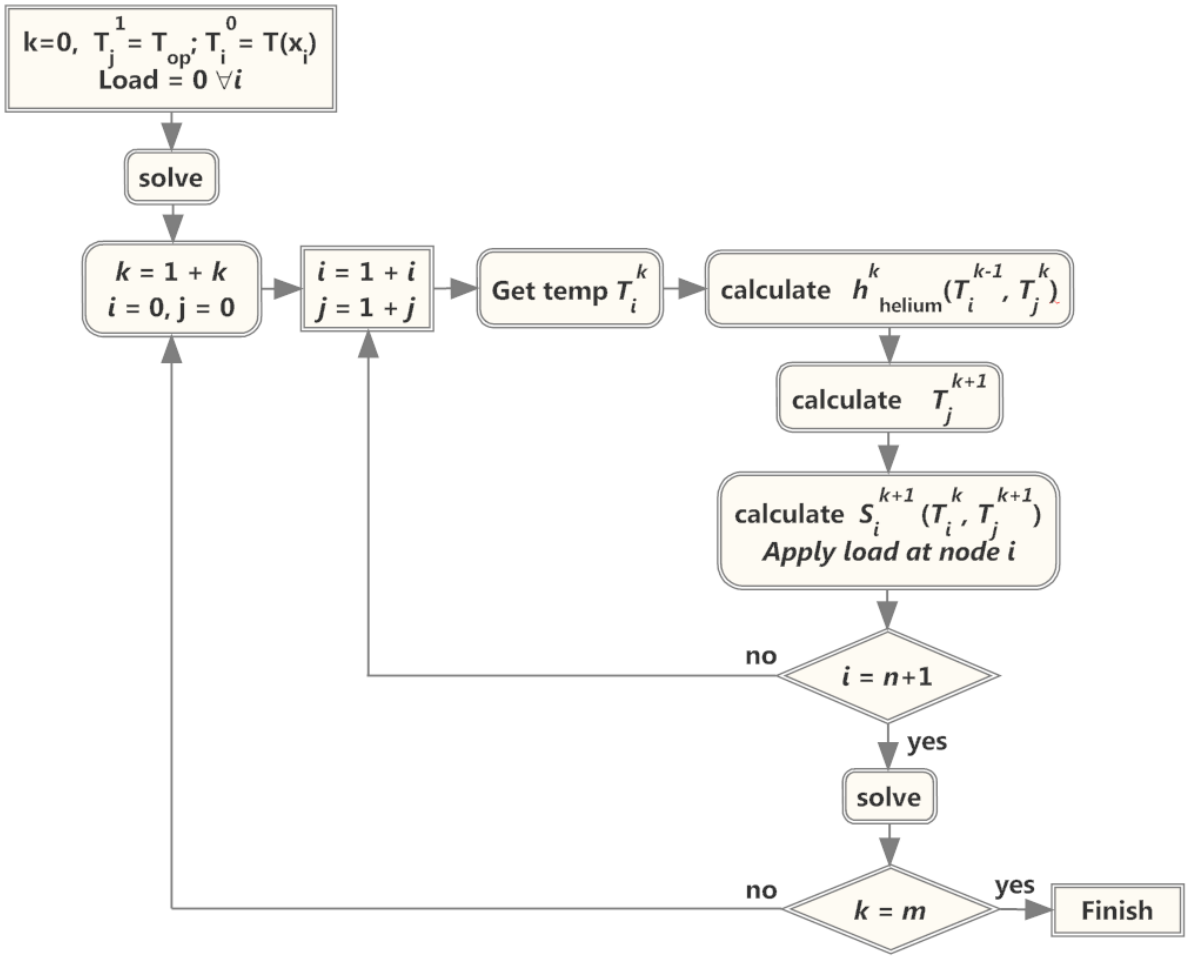


Figure 24 Block diagram of the algorithm that updates the load values to the elements every time steps.

For the initial time step there are no loads. For the rest of the steps, first the temperature of liquid helium is updated then the load vector is updated based on the new temperature of liquid helium, previous temperature of the superconductor and the previous heat flow rate to helium.

4.3.4 Solution and the results

For all of the above models (Model 1-3) transient thermal analysis was carried out from 0 to *t* s. Since we need to update the heat transfer parameters every time step it is not possible to use the adaptive time stepping scheme of ANSYS. Building custom time stepping algorithm was beyond the scope of the thesis therefore we did not go further in programming ourselves and used fixed time stepping scheme. The time step size equal to 10 μs was specified to the solver. Similar to the adiabatic model, the Newton-Raphson iteration scheme was selected to deal with the material non-linearity. Figure 25 shows the results obtained for the setting given in the Table 5.

Table 5 Simulation parameters defined for the quench simulation in helium-cooled models.

Parameter	Symbol	Value	Unit
Discretization	n	1000	-
Transport current	I	150	A
Magnetic field	B	2.88	T
Operating temperature	T_{op}	1.9	K
Hotspot temperature	T_{pick}	20	K
Gaussian parameter	α	0.1	m
Total simulation time	t	0.2	s
Empirical constant	n_{kap}	3	-
HTC (Kapitza regime)	a_{kap}	200	$Wm^{-2} K^{nkap}$
HTC (Film boiling II)	a_{FB-II}	200	$Wm^{-2} K^{-1}$
HTC (Natural convection)	a_{NC}	500	$Wm^{-2} K^{-1}$
HTC (Nucleate boiling)	a_{NB}	50000	$Wm^{-2} K^{-2.5}$
HTC (Film boiling I)	a_{FB-I}	220	$Wm^{-2} K^{-1}$
Heat flux limit (Kapitza regime)	Q_{kap}	35000	Wm^{-2}
Heat flux limit (Natural convection)	Q_{NC}	10	Wm^{-2}
Heat flux limit (Nucleate boiling)	Q_{NC}	20000	Wm^{-2}

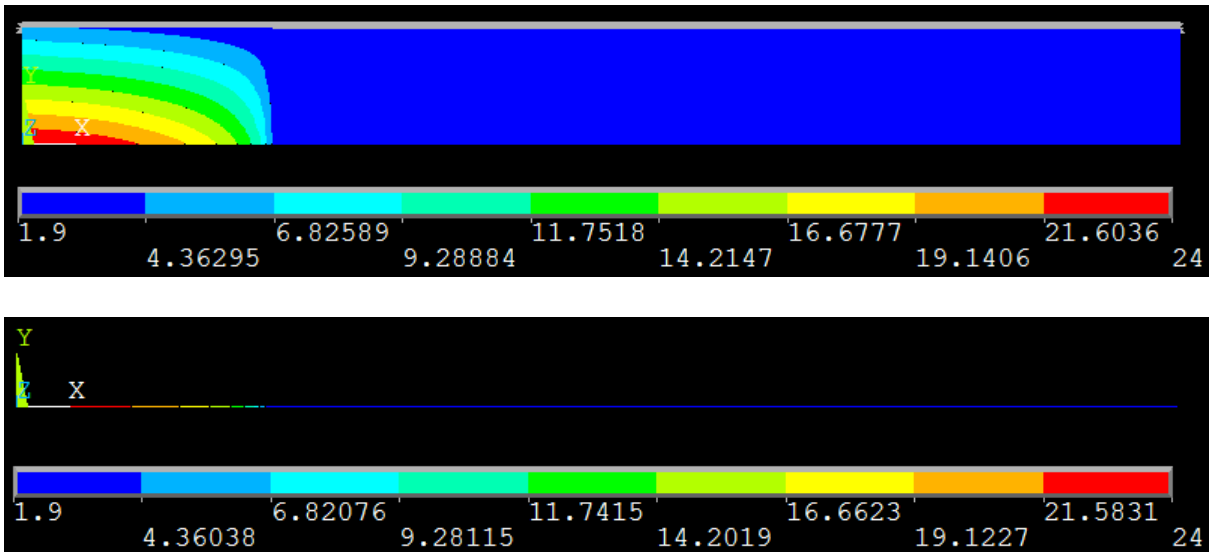


Figure 25 Contour plot of Temperatures (K) obtained for from the analysis for the setting summarised in Table 5, Uppper: From Model 2, Lower: From Model 3.

Table 5 shows that the superconductor domain is discretised in the scale of millimetre. The discretization is so fine that it is difficult to distinguish uniaxial contact elements and the result is obtained as a 2D plot; see upper part of Figure 25. The lower edge of the 2D plot represents the conductor domain and the upper edge represents the helium domain. The temperature

gradients seen in the contact elements is the linearized representation of the temperature difference between two domains.

Figure 26 presents the temperature estimation on the superconducting wire at different times. The curves were obtained from ANSYS Model 1. Similar results were obtained from Model 2 and Model 3 as well. The red curve depicts the imposed initial temperature profile.

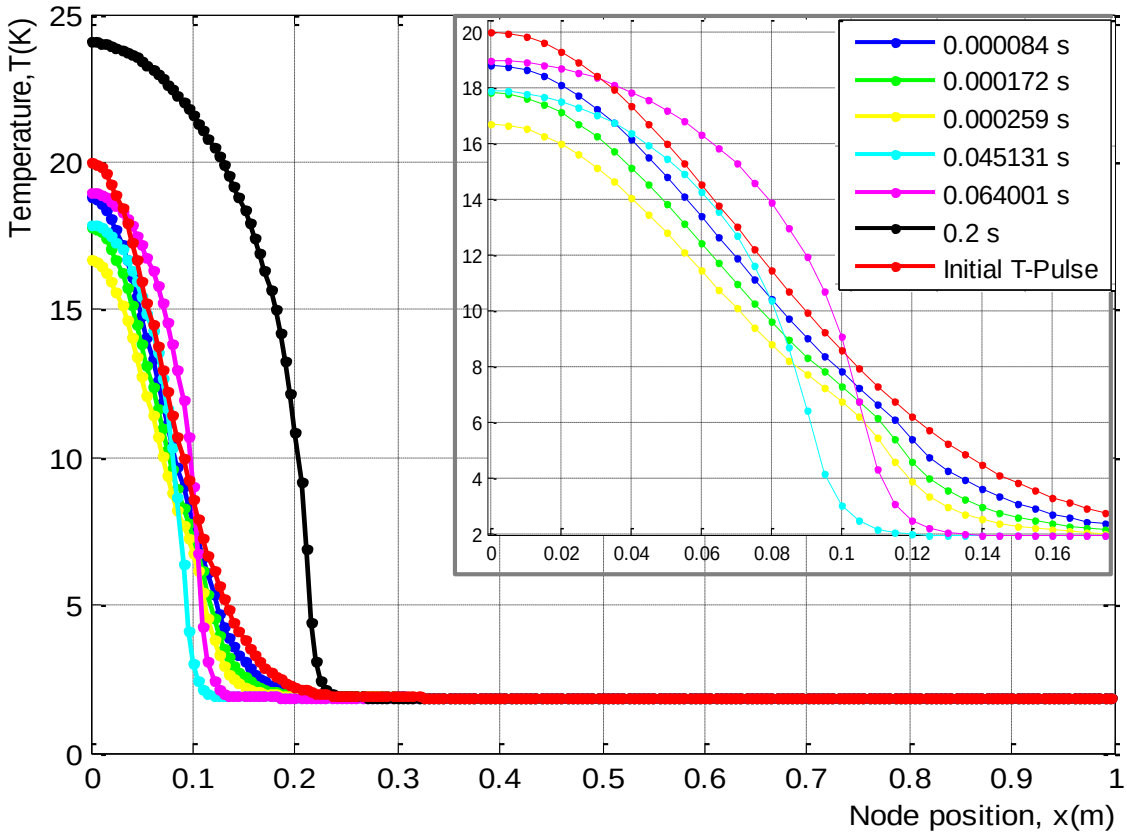


Figure 26 Temperature profiles of the superconducting wire, at different time, estimated by ANSYS Model 1, for the setting summarised in the Table 5.

Similar to the adiabatic model the hot-spot temperature drops initially then as quench propagates the temperature keep on rising until the end of analysis, nevertheless the quench propagation is significantly slow in this case.

Figure 27 compares the temperature on the superconducting wire estimated by ANSYS Models and QP3 for the setting summarised in Table 5. The blue curves represents the imposed initial temperature distribution, and the red curves in the sub-plots is the calculated temperature difference between ANSYS and QP3.

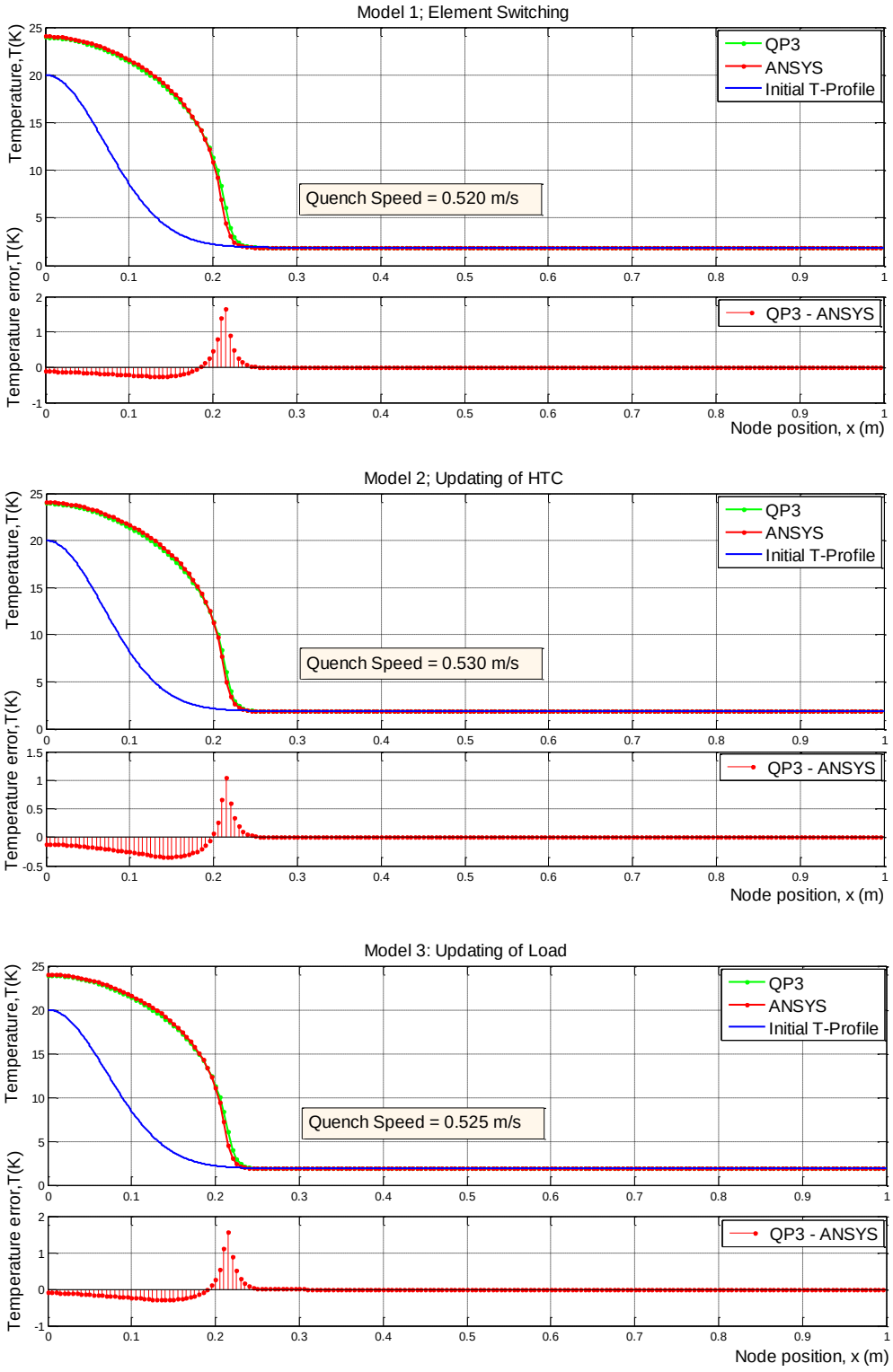


Figure 27 Temperature profile of the superconducting wire estimated by ANSYS Models and QP3 at time *t* = 0.2 s.

The plots show satisfactory results in temperature estimation except the expected significant difference at quench front position. The variation in the estimation of the hot-spot temperature and the quench propagation speed is presented in Table 6.

Figure 28 shows the quench front position at different time. Curves are in reference to plots in Figure 26 and similar plots obtained from other two ANSYS Models and QP3. The quench speed is calculated based on the initial and final position of the quench front.

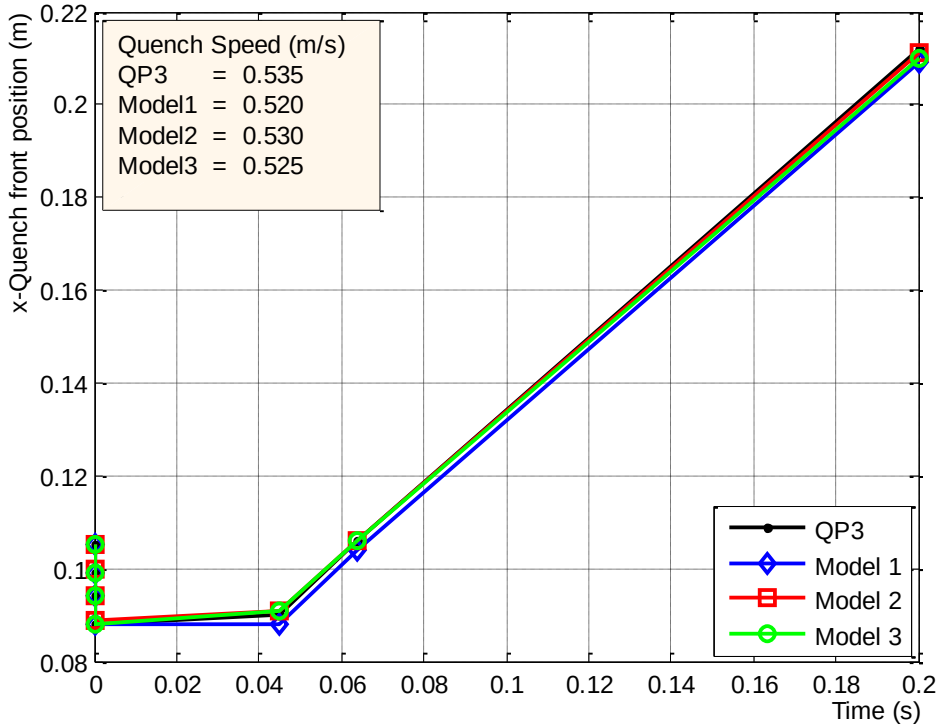


Figure 28 Quench front position with respect to time estimated by ANSYS models and QP3.

Table 6 Variation in the hot-spot temperature and quench propagation speed estimated by ANSYS Models in reference to QP3.

Simulation Environment	Hot-spot temperature difference (K)	Quench propagation speed difference (m/s)
Model 1	0.11 (0.46%)	0.015 (2.80%)
Model 2	0.11 (0.46%)	0.005 (0.93%)
Model 3	0.09 (0.37%)	0.010 (1.87%)

The hot-spot temperature and the quench propagation speed estimated by the models are satisfactory. Table 6 shows that the Model 2 is more accurate among the three different models. One can notice from Figure 16, 27 and 28 that the quench propagation speed estimated by the helium cooled models is nearly ten times slower compared to adiabatic models, whereas, the estimated rise in the hot-spot temperature is nearly half. This suggests that the helium cooling is contributing more in slowing down quench propagation than in taking heat away from the conductor.

The time step size varies depending upon the type of problem. For non-linear problems time step size needs to be very small. We did convergence analysis to find out the minimum time step size for our problem. The results are shown in Figure 29. For the same settings presented in Table 5 the temperature profile at time 0.01 s is plotted for different time step sizes.

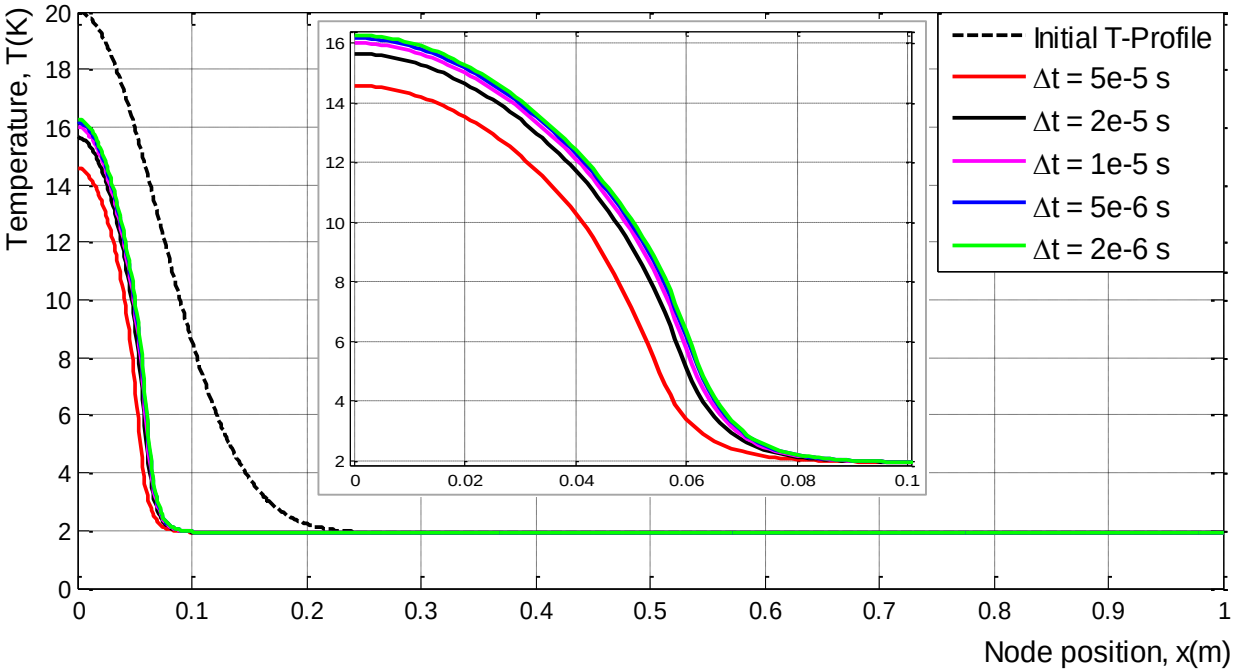


Figure 29 Convergence plot: Temperature profile of the superconducting wire at t = 0.01 s for different time step sizes.

Figure 30 shows the hot-spot temperature (black curve) for different time step sizes. The green curve is the calculated difference in hot-spot temperature between two adjacent time step sizes.

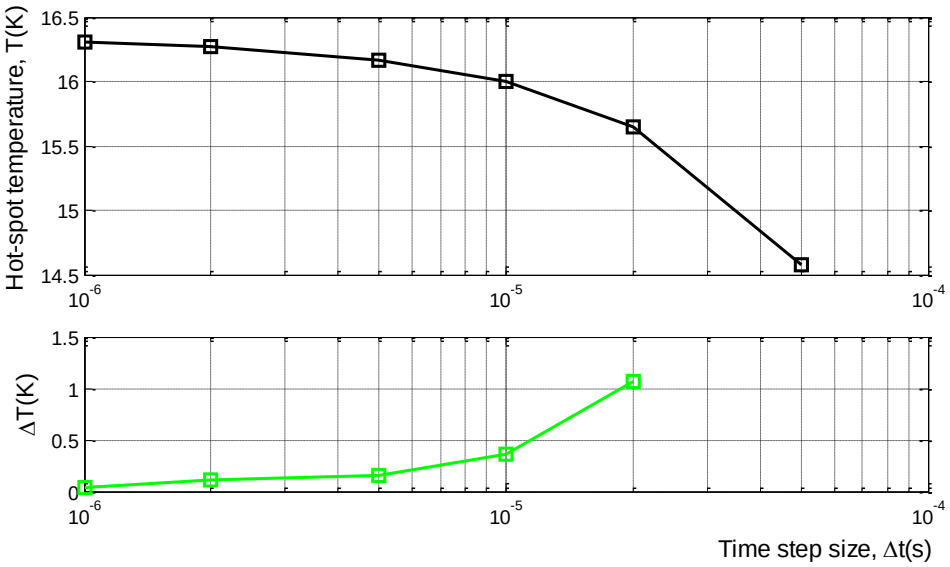


Figure 30 Upper: hot-spot temperature for different time step sizes. Lower: Difference in hot-spot temperature between two adjacent time step sizes.

For the time step size smaller than 10 μs, we achieved the desired accuracy i.e. ≤ 1% change in the hot-spot temperature. The total time taken in ANSYS Model 2 to simulate the phenomena for 0.001 s with time step size equal to 10 μs is 280 s, whereas in QP3 adaptive time stepping method is used where the lower and upper bounds of the time step size are 3 μs and 100 μs respectively and total time taken is 5.5 s.

4.4 Summary of the Results

4.4.1 Adiabatic model

Validation of ANSYS and COMSOL model with QP3 (Figure 16)

In reference to QP3, the error in the hot-spot temperature and quench propagation speed estimated by ANSYS is 0.60% and 1.09% respectively. Similarly, the error by COMSOL is 0.37% and 1.53 % respectively. This shows that both COMSOL and ANSYS are able to calculate hot-spot temperature and quench propagation speed with desired accuracy, i.e, < 1% error in hot-spot temperature and < 2% error in quench propagation speed.

Convergence analysis with respect to mesh refinement (Figure 17).

For both ANSYS and COMSOL a converged solution was obtained for spatial discretization in the scale less than or equal to a millimetre for which the change in quench speed dropped below 1%. The converged solution of ANSYS and COMSOL varies by 0.4 % in hot-spot temperature and by 2.49% in quench propagation speed. The required scale of mesh refinement, i.e, a millimetre scale is a well-known empirical value. The results obtained from the convergence analysis is as expected.

4.4.2 Helium-cooled model

Validation of ANSYS model with QP3 (Figure 27, Figure 28)

In comparison to QP3, the error in hot-spot temperature and the quench speed estimated by the most accurate ANSYS model is 0.46 % and 0.93% respectively. This shows that ANSYS is able to calculate hot-spot temperature and quench propagation speed with relative accuracy.

Convergence analysis with respect to time step size (Figure 30)

A converged solution was obtained for time step size less than or equal to 10 μs for which the change in hot-spot temperature dropped below 1%. The minimum required time step size, i.e, 10 μs is a well-known empirical value. The result obtained from convergence analysis is as expected.

4.4.3 Performance Analysis

For the mesh refinement corresponding to the converged solution, i.e, 1 mm, the adiabatic analysis is nearly 5 times faster in commercial software (Figure 18). The helium-cooled models, not being able to incorporate adaptive time stepping method, are far more time consuming than QP3.

5. CONCLUSIONS

The main aim of the work was to solve the thermal problem in quench using commercial FEA tools and to compare the results with custom software in order to find out the most efficient method of quench simulation. Modelling of the thermal phenomena in a quench comprehend two main challenges i) the heat generation load and the material properties have non-linear dependency on temperature and magnetic field ii) the heat transfer between the conductor and helium goes through different transfer and boiling regimes depending upon temperatures, heat flux and integrated heat.

In the beginning of the thesis project different approaches to quench simulation were studied in the literature [10-16, 22-24 and 35-36]. The differential heat diffusion equation and the empirical heat transfer equation representing the thermal problem in a quench are formulated in Chapter 2. The practical and theoretical information about quench simulation are summarized in Chapter 3. Two types of numerical model, adiabatic and the one including helium, were constructed using commercial simulation environments. The adiabatic model is based on the differential equation and solves the thermal problem in the superconductor. The helium-cooled model is based on both the differential equation and the empirical equation, and solves the thermal problem in a quench. Chapter 4 presents the models and the analysis carried out to validate the results with custom software, i.e, QP3. The stability and convergence of the models were checked for different spatial and temporal discretization. Based on the observed results and the initial literature review we can now make the following conclusions.

The Problem arising from the non-linear material properties and temperature dependent load was easily solved using ANSYS and COMSOL. The simulation environments included a feature where one can create non-linear material tables and temperature-dependent loads, and provided a straight forward solution to build the adiabatic models. The adiabatic models estimated the hot-spot temperature and quench propagation speed with desired accuracy and were found to be stable for a large number of elements.

Nevertheless, modelling of the heat transfer between the conductor and the helium was difficult in both ANSYS and COMSOL. They did not provide a feature to directly implement the empirical heat transfer equation. To define such equation a separate algorithm was constructed using APDL scripts in ANSYS. The algorithm needs to be executed every time step to determine the applicable regime of helium state and the adaptive time stepping scheme of the simulation environment therefore cannot be used. Implementing a self-made time stepping

algorithm was beyond the scope of the thesis. A fixed time stepping scheme was selected which resulted the helium-cooled model to be far more time consuming than QP3. The models however estimated the hot-spot temperature and the quench propagation speed with satisfactory accuracy.

The thermal phenomena in a quench are correctly simulated using commercial FEA tools. Similar results are reported in [11-13], where the analysis is limited to the adiabatic scenario. The coupled analysis presented in this thesis is a new method to take into account the non-linear cooling behaviour of helium.

6. RECOMMENDATIONS FOR FUTURE WORK.

One could explore the user programmable feature in ANSYS and MatLab/COMSOL, to see if it is possible to reconcile the regime switching algorithm with the automatic time stepping scheme of the simulation environment.

In order to simulate quench propagation in other directions than the longitudinal, one needs to build 2D/3D model representing a magnet coil. The 1D model constructed in ANSYS is based on the finite element method and one can easily modify it to build 2D/3D model. Apart from the superconductor, the conductor, and the helium, a magnet coil consists of insulation material. One can use contact elements to represent the insulation between the cable domains in the same way as was used to model non-linear heat transfer between the conductor and helium; see Appendix 13.

For the simulation of large 3D models, one can explore the use of in-build adaptive spatial discretization method. In ANSYS the method is limited to static analysis. COMSOL however provides the feature for transient analysis as well. It is recommended to check how effectively the method can be implemented and how much is the contribution in reducing the computation cost.

7. REFERENCES

[1] P. Bauer, Stability of Superconducting Strands for Accelerator Magnets. PhD thesis, Technical University Vienna, 1996.

[2] M. N. Wilson, Superconducting Magnets, Oxford University Press, USA, 1990.

[3] K.H. Mess, Superconducting accelerator magnets, World Scientific press, Singapore, 1996.

[4] AP. Verweij, Electrodynamics of Superconducting Cables in Accelerator Magnets, PhD thesis, University of Twente Enschede, 1995.

[5] O. Bruning et al., LHC Design Report, Scientific Information Service, Geneva, 2004.

[6] G. Willering, Stability of Superconducting Rutherford Cables. PhD thesis, University of Twente Enschede, 2009.

[7] P. P. Granieri, Heat Transfer between the Superconducting Cables of the LHC Accelerator Magnets and the Superfluid Helium Bath. PhD thesis, EPFL Lausanne, 2012.

[8] A. Verweij, QP3 User's Manual, CERN EDMS: 1150045, 2008.

[9] L. Bottura et al., THEA Thermal, Hydraulic and Electric Analysis of Superconducting Cable. CryoSoft, Geneva, 2003.

[10] S.Russenschuck, Field Computation for Accelerator Magnets, Strauss –GmbH, Moerlenbach, 2008.

[11] S. Caspi et al., Calculating quench propagation with ANSYS, *IEEE Transactions on Applied Superconductivity, 13(2):1714–1717*, 2003.

[12] R. Yamada et al., 2D/3D quench simulation using ANSYS for epoxy impregnated NB3Sn high field magnets, *IEEE Transactions on Applied Superconductivity, 13(2):1696-1670*, 2003.

[13] G. Volpini. Quench propagation in 1-d and 2-d models of high current superconductors, *Proceedings of the COMSOL Conference Milan,* 2009.

[14] K.B. Kumar et al., Novel techniques to solve sets of coupled differential equations with SPICE, *IEEE Transactions on Circuits and Devices Magazine, 7(1):11-14*, 1991.

[15] G. J. C. Aird et al., Coupled Transient Thermal and Electromagnetic Finite Element Simulation of Quench in Superconducting Magnets, *Proceedings of ICAP Chamonix,* 2006.

[16] S.Posen et al., Copuled Electromagnetic-Thermal-Mechanical Simulations of Superconducting RF Cavities, *Proceedings of IPAC Kyoto*, 2010.

[17] M.S. Lubell et al., Empirical scaling formulas for critical current and critical field for commercial NbTi, *IEEE Transactions on magnetics, 19(3):754-757*, 1983.

[18] T.Schreiner, Current distribution inside Rutherford-type superconducting cables and impacto on performance of LHC dipoles, PhD thesis, Technical University Viena, 2002.

[19] E.R. Bielert et al., Implementation of the superfluid helium phase transition using finite element modelling: Simulation of transient heat transfer and He-I/He-II phase front movement in cooling channels of superconducting magnets, *ELSEVIER Cryogenics 53:78-85*, 2013.

[20] S.W. Van Sciver, Helium Cryogenics, Springer, New York, 2012.

[21] Michal R.Gosz, Finite Element Application in solids, structures and heat transfer, CRC press, London, 2006.

[22] N. Schwerg et al., Quench Simulation in an Integrated Design Environment for Superconducting Magnets, *IEEE Transactions on Magnetics, 44(6):934-937*, 2008.

[23] M.Wake et al., "Complete Quench Simulation of Large Solenoid Magnet", *IEEE Transactions on Applied Superconductivity, 22(3)*, 2012.

[24] V.Zermeno, Computation of Superconducting Generators for Wind Turbine Applications, PhD thesis, Technical University of Vienna, 2012.

[25] M.A. Crisfield, Non-Linear Finite Element Analysis of Solids and Structures, John Wiley & Sons, New York, 1997.

[26] O. C. Zienkiewicz, Finite element method, its basics and fundamentals, ELSEVIER, USA, 2001.

[27] B. Auchmann, Numerical Treatment of Eddy Currents in LHC Dipole Magnets Using BEM-FEM Coupling, Master Thesis, Wien Bergheideng, 2001.

[28] J.C.Sanders, Nonlinear, Transient Conduction Heat Transfer Using a Discontinuous Galerkin Hirarchical Finite Element Method, Master Thesis, 2004.

[29] G.Noh et al., An explicit time integration scheme for the analysis of wave propagations, *ELSEVIER computer and structures, 129:178-193,* 2013

[30] M. O. Domingues et al., An adaptive multiresolution scheme with local time stepping for evolutionary PDEs, *Journal of computation physics, 227:3758–3780,* 2008.

[31] ANSYS Documentation, https://support.ansys.com, Last checked Dec. 2014.

[32] V.LE. Canh, Novel Numerical Procedures for limit analysis of structures, PhD thesis, University of Sheffield, 2009.

[33] B. Auchmann, The Coupling of Discrete Electromagnetism and the Boundary Element Method for the Simulation of Accelerator Magnets, PhD thesis, Wien Bergheideng, 2004.

[34] H.S.R.Rondan, Material Laws and Numerical Methods in Applied Superconductivity, PhD thesis, University of Zaragoza Spain, 2012.

[35] V. Bansal et al., 3D Design, Electro-Thermal Simulation and Geometrical Optimization of spiral Platinum Micro-heaters for Low Power Gas sensing applications using COMSOL, *Proceedings of the 2001 COMSOL Conference in Bangalore,* 2001.

[36] D. Hu et al., 3D Modelling of All-Superconducting Synchronous Electric Machines by the Finite Element Method, *Proceedings of the 2014 COMSOL Conference in Cambridge,* 2014.

[37] K. Dahlerup Pettersen et al., The Protection System for the Superconducting Elements of the Large Hadron Collider at CERN, *Proceedings of the Particle Accelerator Conference New York,* 1999.

[38] R. Denz et al., Electronic Systems for the Protection of Superconducting Elements in the LHC, *IEEE Transactions on Applied Superconductivity, 16(2):1725-1728,* 2006.

[39] E. Fornasiere et al., Status of the Activities on the Nb_3Sn Dipole SMC and of the Design of the RMC, *IEEE Transactions on Applied Superconductivity, 23(3),* 2013.

List of Figures

List of Tables

APPENDICES

1. Analytical Model

To simplify the problem let us consider an adiabatic case, i.e. no helium cooling and no heat goes in and out of the magnet coil. The relation between the total Joule heat and the temperature rise in the coil can be expressed by time dependent heat balance equation [2 and 10].

$$\frac{\rho_{\mathrm{el}}}{a_{\mathrm{cu}}} I(t)^2 = c_{\mathrm{p}}(T) a_{\mathrm{cable}} \frac{dT}{dt}. \tag{1}$$

Where ρ_{el} is the electrical resistivity of copper. a_{cu} is the copper cross section, a_{cable} total cross section, $I(t)$ the current and $c_{\mathrm{p}}(T)$ the temperature dependent equivalent heat capacity of the cable. The resistivity of Nb-Ti superconductor is orders of magnitude less than that of copper at normal conducting state so its resistivity can be neglected. Assuming that the quench starts at $t = 0$ s with initial temperature T_0 and the temperature at time t is T, the separation of variables and integration yields [10]

$$\int_0^t I(t)^2 \mathrm{d}t = a_{\mathrm{cu}} a_{\mathrm{cable}} \int_{T_0}^{T} \frac{C_{\mathrm{p}}(T)}{\rho_{\mathrm{el}}} \mathrm{d}T \ . \tag{2}$$

The term on the left hand side of the equation, the time integral of the square of the current, is called MIIts and is usually expressed in units of 10^6 A^2s [10]. It represents the quench load and can be determined by measuring the current decay of the magnet during the quench. Using (2) we can roughly estimate the average temperature of the coil during the quench (it is a worst-case scenario).

2. Quench detection and Magnet protection

A quench is detected by measuring the voltage across the magnet. The voltage measured by voltage taps in the magnet includes the inductive (because of the changing current in the magnet over time) and resistive voltage. Thus, the detection system should be able to subtract the inductive voltage from the measurement in order to obtain resistive voltage component [37 and 38].

Magnet protection mainly consists of two methods: i) passive protection ii) active protection. In passive protection scheme a diode or a resistor is connected to the magnet which provides parallel path to the current. The scheme works for the magnet persisting cryogenic stabilization even after the quench. The cryogenic stabilization means either the matrix material is sufficient

enough to withstand the current decay without overheating or the Ohmic heating in the copper matrix is smaller than the rate of heat transfer to helium bath [37 and 38].

In active protection scheme once the quench is detected an appropriate protection system is triggered. The current decay is speeded up by extracting energy into the dump resistor and/or quench heaters are fired at several location of the magnet to speed up the quench propagation. The stored energy in the coil is dissipated either outside of magnet or over the larger fraction of magnet coil thus limiting the increase of the peak temperature [10].

3. Heat transfer in bulk helium

The heat transfer in the bulk helium can be explained by two regimes: laminar (Landau) regime and turbulent (Gorter-Mellink) regime. Below a certain limiting heat flux there exists a laminar friction-less counter-flow between the normal component and the superfluid component of helium. The non-viscous superfluid component tends to move towards the heated surface, while normal component moves away taking the heat along with it. Upon crossing the limit of heat flux the relative velocity of counter-flow exceeds the critical velocity. The counter-flow being no longer frictionless becomes turbulent [19].

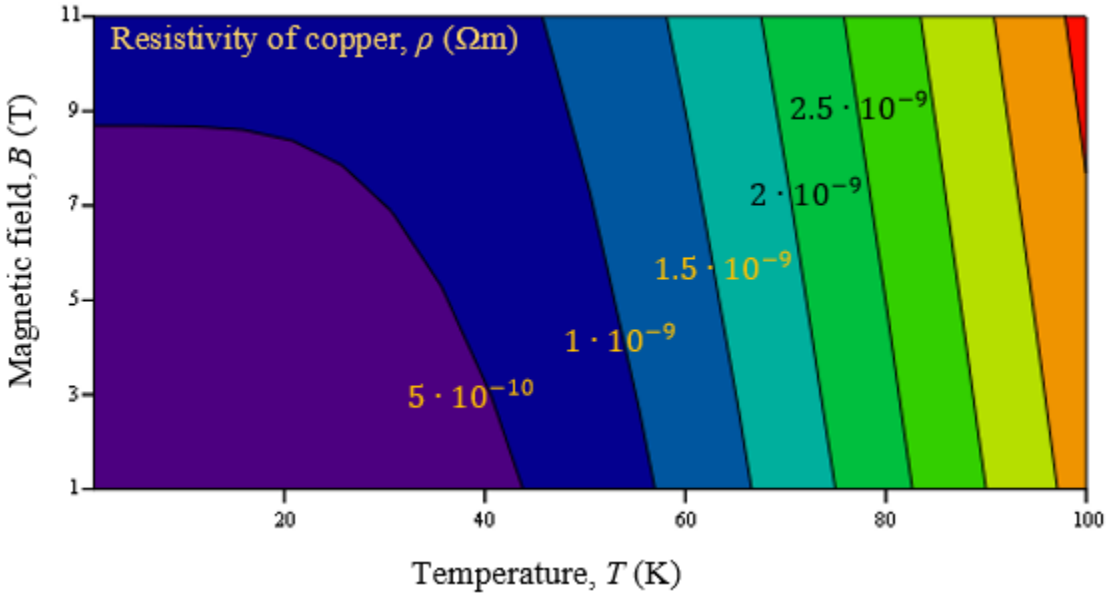
4. Non-linear inductance induced losses

"For the LHC main dipoles it takes about 1200s to reach from the injection field of 0.58 T to the nominal field of 8.4 T" [4]. The changing magnetic field with respect to time during the up/down ramp produces coupling currents at contact resistances inside the cables and eddy currents in the copper spacers, collars and yokes. The coupling currents are: IFCCs (Interfilament coupling current), ISCCs (Interstrand coupling current) and BICCs (Boundary-induced coupling currents). ISCCs induces joule heating in the contact resistance and IFCCs induces joule heating in the copper matrix. If the heat generated raises the temperature above the current-sharing temperature quench initiates.

5. Resistivity of Cu

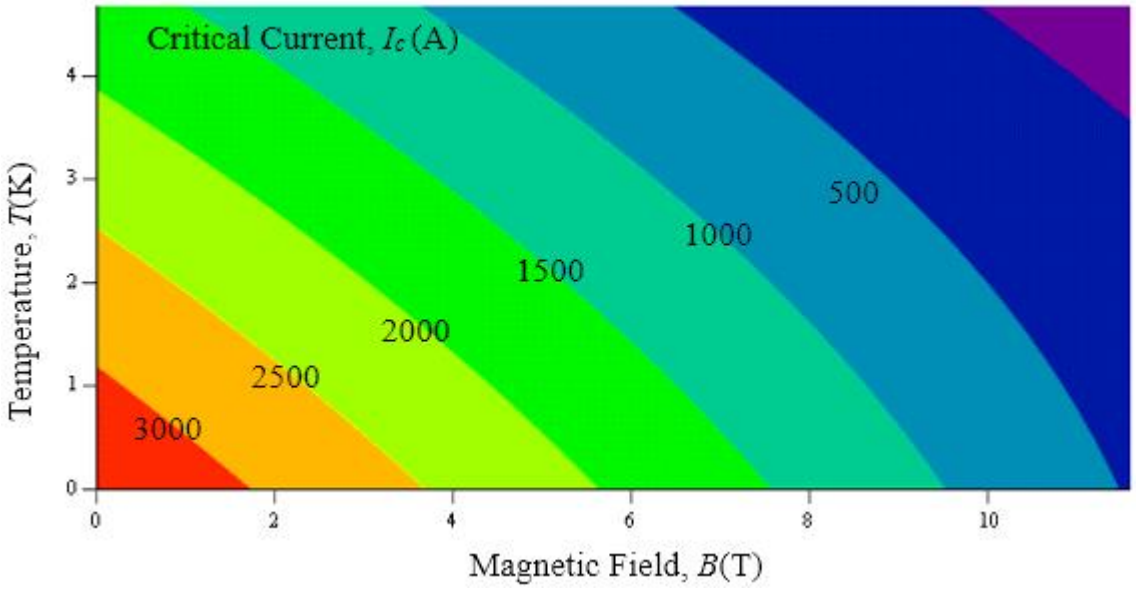
$$\rho(T,B) := \left[\frac{1.7}{RRR} + \frac{1}{\left(\frac{2.33\times 10^{9}}{T^{5}} + \frac{9.57\times 10^{5}}{T^{3}} + \frac{163}{T}\right)}\right]\cdot 10^{(-8)} + (0.37 + 0.0005RRR)\cdot B\cdot 10^{(-10)}\,(\Omega\cdot m)$$

For residual resistivity ratio, RRR = 150.



6. Critical Current for Nb-Ti Superconductor

$$I_c(T,B) := (3449 - 257\cdot B)\cdot\left[1 - \frac{T}{9.2\left(1 - \frac{B}{14.5}\right)^{0.59}}\right](A)$$

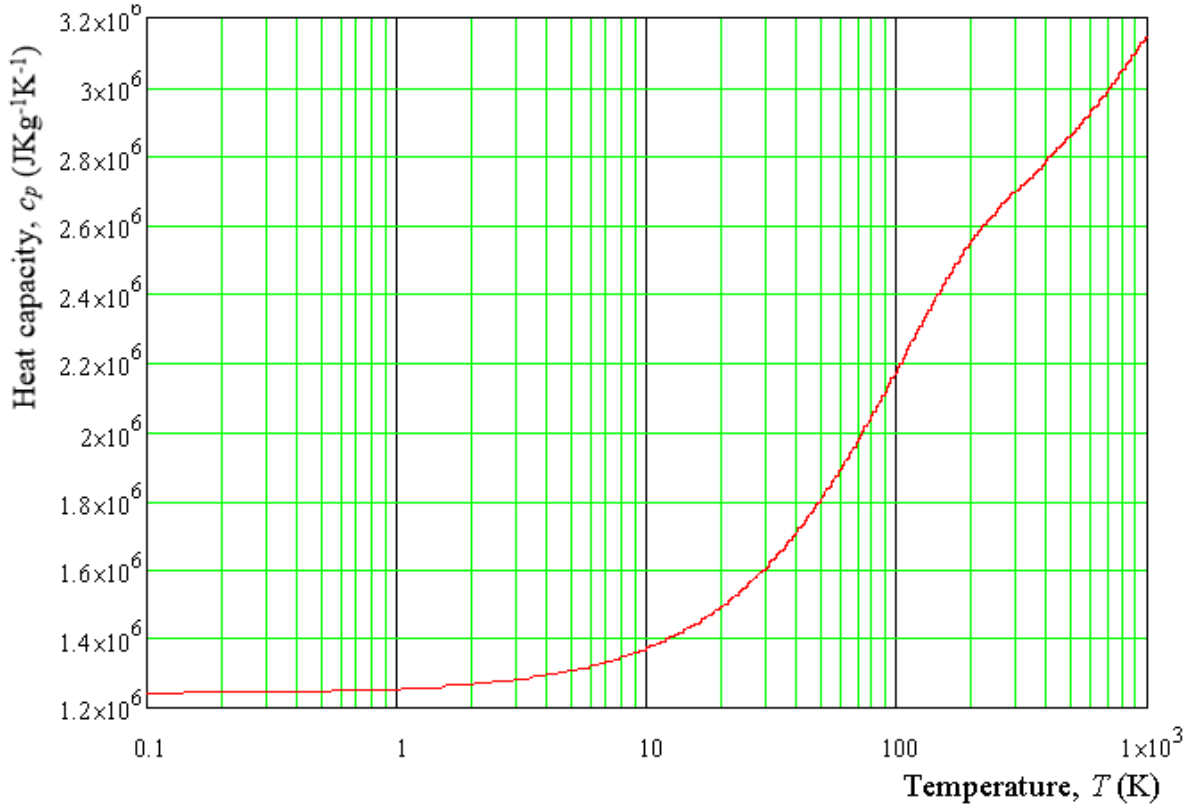


7. Heat capacity of Nb-Ti

$$cpnbti(T, Tc) := a \cdot T^4 + b \cdot T^3 - c \cdot T^2 + d \cdot T^1 + e \qquad [Jkg^{-1}K^{-1}]$$

	a	b	c	d	e
for $T<T_c$	0	4.910E+01	0	6.400E+01	0.000E+00
for $T_c<T<20$ K	0	1.624E+01	0	9.280E+02	0.000E+00
for 20 K$<T<$50 K	-2.177E-01	1.198E+01	5.537E+02	-7.846E+03	4.138E+04
for 50 K$<T<$175 K	-4.820E-03	2.976E+00	-7.163E+02	8.302E+04	-1.530E+06
for 175 K$<T<$500 K	-6.290E-05	9.296E-02	-5.166E+01	1.3706E+04	1.240E+06
for 500 K$<T<$1000 K	0	0	-2.570E-01	9.555E+02	2.450E+06

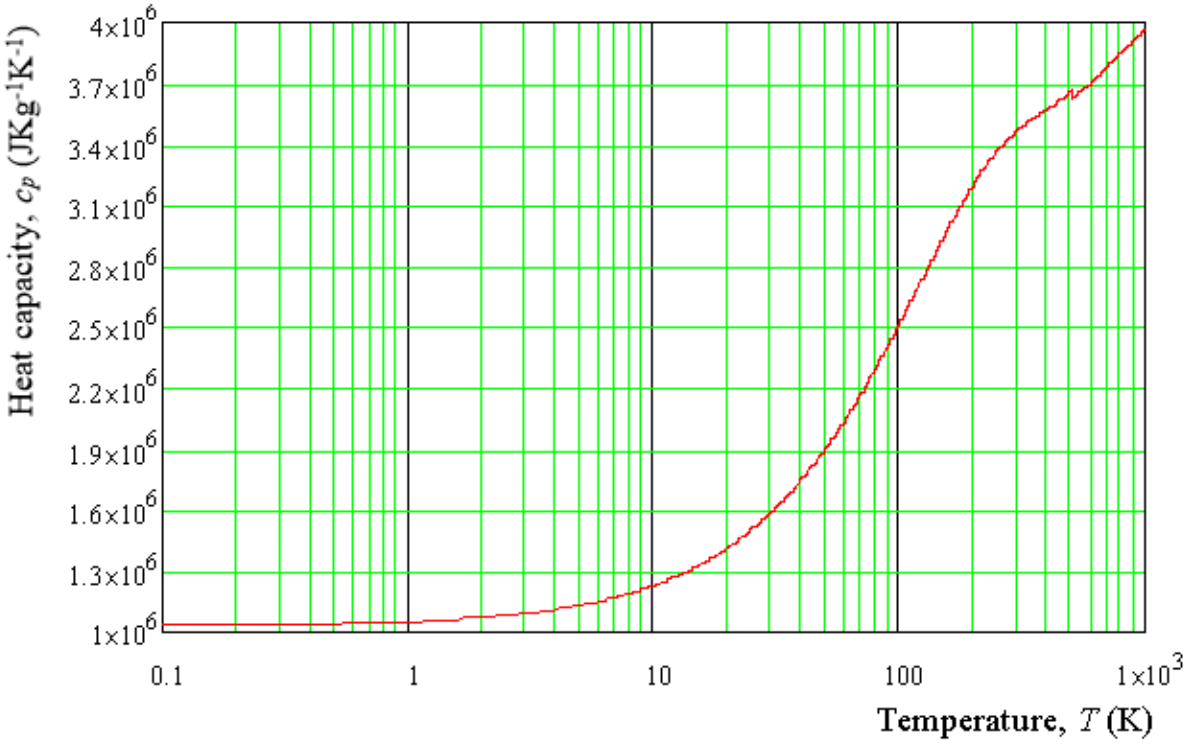
For critical temperature, T_c = 8.07 K.



8. Heat capacity of Cu

$$cpcu(T,Tc) := a\cdot T^4 + b\cdot T^3 - c\cdot T^2 + d\cdot T^1 + e \qquad [Jkg^{-1}K^{-1}]$$

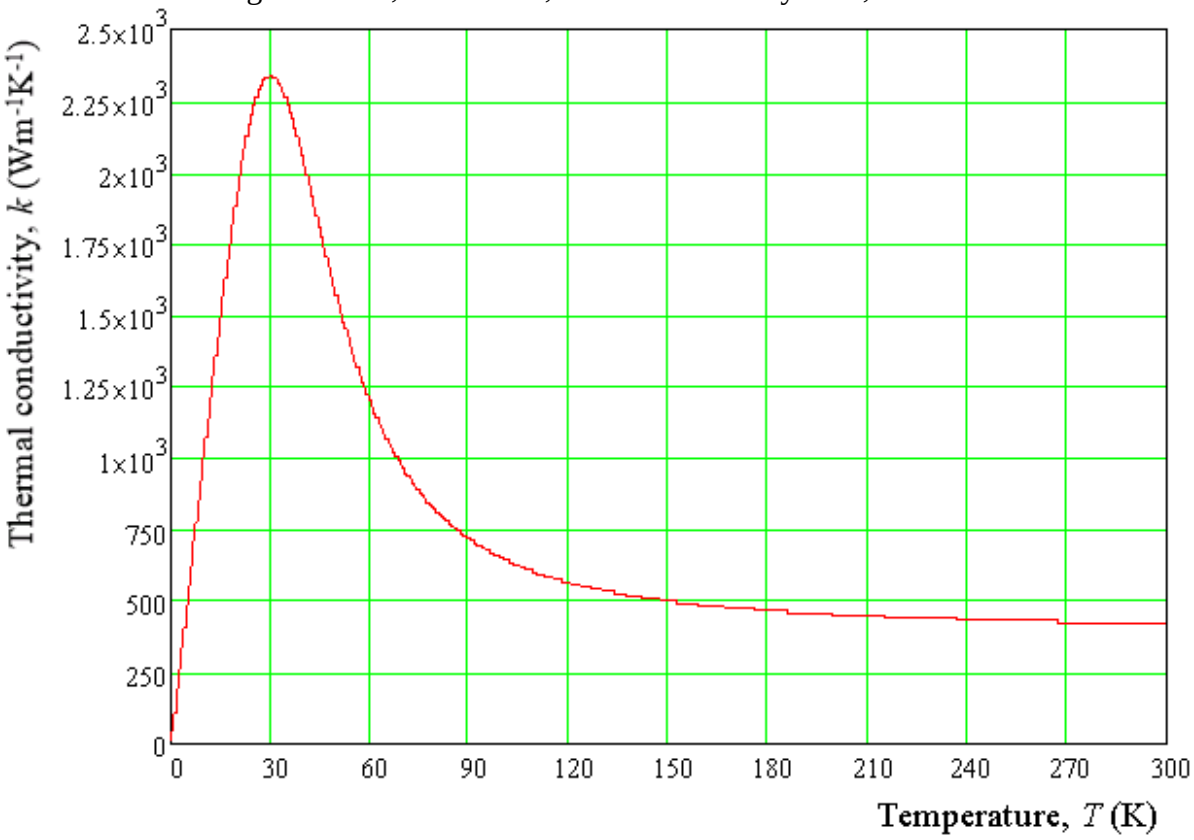
	a	b	c	d	e
for ***T*<10 K**	-3.080E-02	7.230E+00	-2.129E+00	1.019E+02	2.563E+00
for **10 K<*T*<40 K**	-3.045E-01	2.987E+01	-4.556E+02	3.470E+03	-8.250E+03
for **40 K<*T*<125 K**	4.190E-02	-1.402E+01	1.509E+03	-3.160E+04	1.784E+05
for **125 K<*T*<300 K**	-8.480E-04	8.419E-01	-3.255E+02	6.059E+04	-1.290E+06
for **300 K<*T*<500 K**	-4.800E-05	9.173E-02	-6.412E+01	2.036E+04	1.030E+06
for 5**00 K<*T*<1000 K**	0.000E+00	1.200E-05	-2.149E-01	1.004E+03	3.180E+06



9. Thermal conductivity of Cu

$$k_{Cu}(T,B) := \frac{2.45\,10^{-8}\cdot T}{\left[\frac{1.7}{RRR} + \frac{1}{\left(\frac{2.33\times 10^{9}}{T^{5}} + \frac{9.57\times 10^{5}}{T^{3}} + \frac{163}{T}\right)}\right]\cdot 10^{(-8)} + (0.37 + 0.0005RRR)\cdot B\cdot 10^{(-10)}}\ W\cdot m^{-1}K^{-1}$$

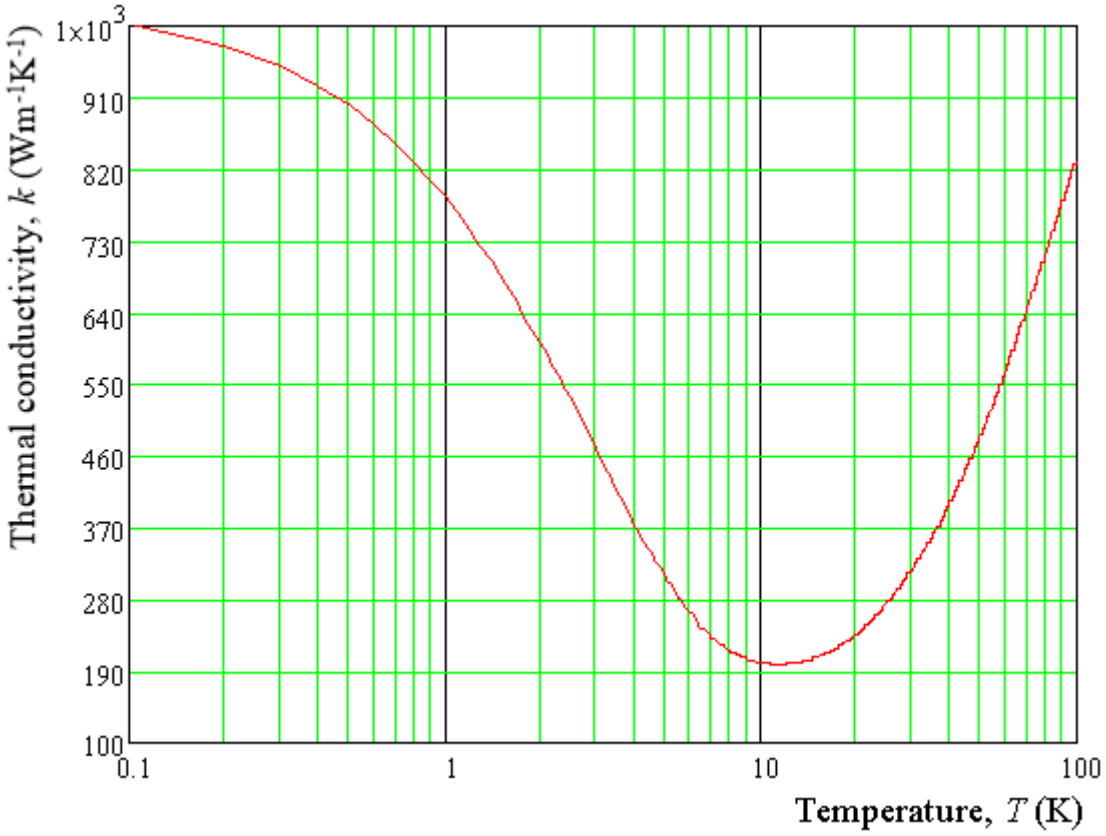
For magnetic field, B = 2.88 T, residual resistivity ratio, RRR = 150.



10. Thermal conductivity of He

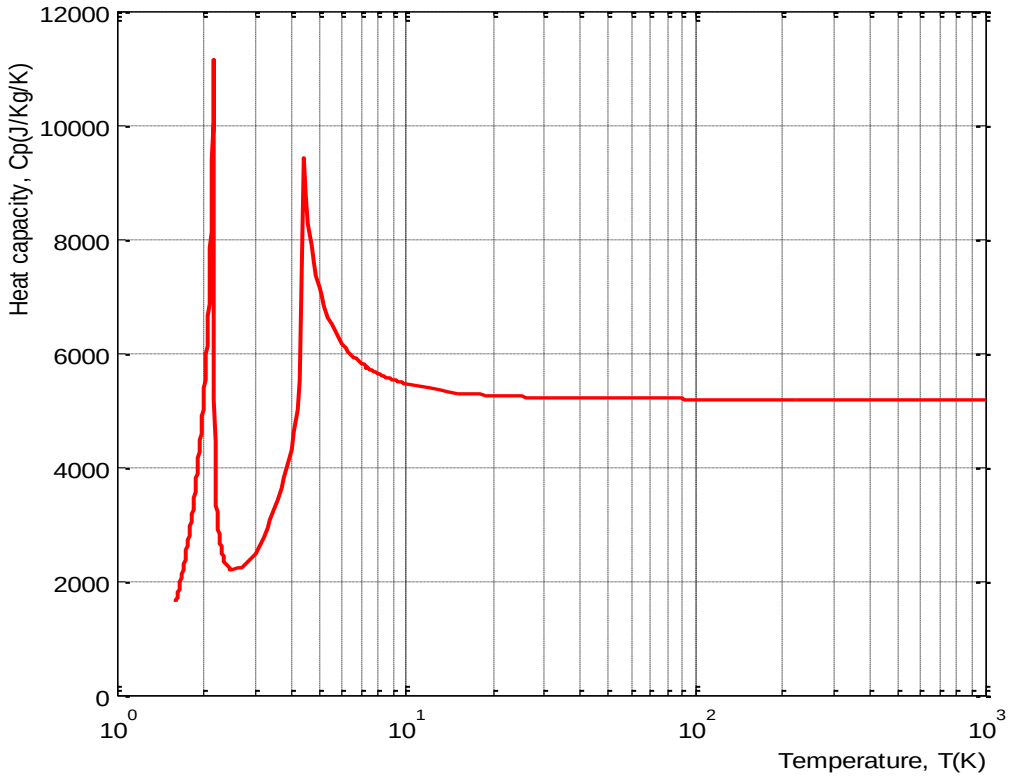
$$khe(T) := a \cdot T^{6} + b \cdot T^{5} + c \cdot T^{4} + d \cdot T^{3} + e \cdot T^{2} + f \cdot T^{1} + g \qquad [Wm^{-1}K^{-1}]$$

	a	b	c	d	e	f	g
for ***T*<0.3 K**	0	0	0	0	0	0	2.48E+03
for **0.3 K<*T*<8 K**	2.16E-01	-6.94E+0	8.90E+01	-5.81E+2	2.05E+3	-3.79E+3	3.45E+03
for **8 K<*T*<20 K**	0	-1.21E-3	8.81E-2	-2.56E+0	3.76E+01	-2.79E+2	1.03E+03
for **20 K<*T*<1000 K**	2.02E-11	-2.11E-8	8.53E-06	-1.63E-3	1.22E-01	4.80E+00	1.00E+02



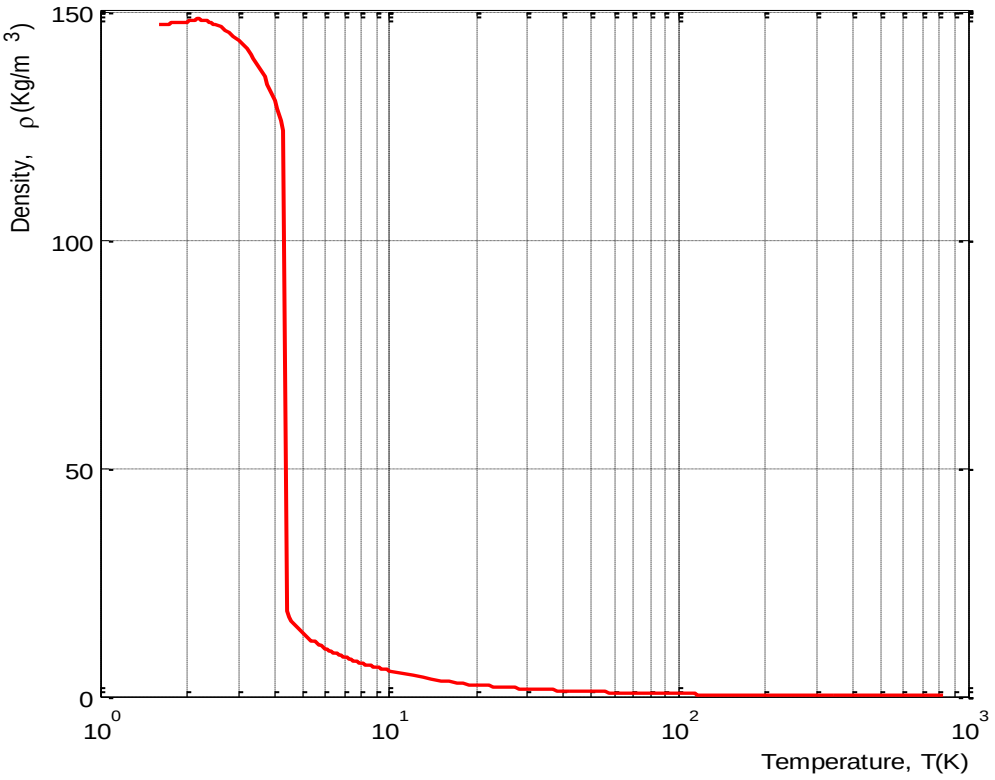
11. Heat capacity of He

T(K)	Cp(J/kg/K)	T	Cp	T	Cp	T	Cp	T	Cp
1.6	1616	1.944	4457.8	2.104	7552.3	2.264	2758.9	2.424	2237.5
1.7	2210.1	1.952	4562	2.112	7826	2.272	2701.8	2.432	2228.8
1.8	2962.5	1.96	4669.2	2.12	8130.8	2.28	2650.8	2.44	2220.9
1.808	3031	1.968	4779.7	2.128	8477.1	2.288	2605.1	2.448	2213.8
1.816	3100.9	1.976	4893.6	2.136	8882.3	2.296	2564	2.456	2207.5
1.824	3172.2	1.984	5011.2	2.144	9379.4	2.304	2526.8	2.464	2201.8
1.832	3245	1.992	5132.7	2.152	10044	2.312	2493.1	2.472	2196.7
1.84	3319.3	2	5258.5	2.16	11139	2.32	2462.4	2.48	2192.3
1.848	3395.3	2.008	5388.9	2.168	7817.4	2.328	2434.5	2.488	2188.4
1.856	3472.9	2.016	5524.3	2.176	5185.5	2.336	2409	2.496	2191
1.864	3552.2	2.024	5665.1	2.184	4440	2.344	2385.7	2.5	2192.9
1.872	3633.3	2.032	5811.9	2.192	4013	2.352	2364.4	2.6	2221.2
1.88	3716.2	2.04	5965.1	2.2	3721.1	2.36	2344.8	2.7	2245.5
1.888	3801.1	2.048	6125.6	2.208	3503.7	2.368	2326.9	2.8	2292.7
1.896	3888	2.056	6294.2	2.216	3333.1	2.376	2310.5	2.9	2370.5
1.904	3977.1	2.064	6471.8	2.224	3194.5	2.384	2295.5	3	2479.3
1.912	4068.3	2.072	6659.8	2.232	3079.1	2.392	2281.7	3.1	2615.4
1.92	4161.9	2.08	6859.5	2.24	2981.2	2.4	2269.1	3.2	2763.7
1.928	4257.9	2.088	7073	2.248	2896.9	2.408	2257.6	3.3	2919.7
1.936	4356.5	2.096	7302.8	2.256	2823.5	2.416	2247.1	3.4	3081.9



12. Density of He

T(K)	ρ(Kg/m^3)	T	ρ	T	ρ	T	ρ	T	ρ
1.6	147.25	1.76	147.38	1.92	147.59	2.08	147.95	2.24	148.23
1.608	147.2556	1.768	147.388	1.928	147.6	2.088	147.97	2.248	148.21
1.616	147.2612	1.776	147.396	1.936	147.62	2.096	148	2.256	148.18
1.624	147.2668	1.784	147.404	1.944	147.63	2.104	148.03	2.264	148.16
1.632	147.2724	1.792	147.412	1.952	147.65	2.112	148.05	2.272	148.14
1.64	147.278	1.8	147.42	1.96	147.66	2.12	148.08	2.28	148.11
1.648	147.2836	1.808	147.43	1.968	147.68	2.128	148.11	2.288	148.08
1.656	147.2892	1.816	147.44	1.976	147.69	2.136	148.15	2.296	148.06
1.664	147.2948	1.824	147.45	1.984	147.71	2.144	148.18	2.304	148.03
1.672	147.3004	1.832	147.46	1.992	147.73	2.152	148.22	2.312	148
1.68	147.306	1.84	147.47	2	147.74	2.16	148.27	2.32	147.97
1.688	147.3116	1.848	147.48	2.008	147.76	2.168	148.32	2.328	147.94
1.696	147.3172	1.856	147.49	2.016	147.78	2.176	148.33	2.336	147.91
1.704	147.324	1.864	147.51	2.024	147.8	2.184	148.33	2.344	147.88
1.712	147.332	1.872	147.52	2.032	147.82	2.192	148.32	2.352	147.85
1.72	147.34	1.88	147.53	2.04	147.84	2.2	148.31	2.36	147.82
1.728	147.348	1.888	147.54	2.048	147.86	2.208	148.3	2.368	147.79
1.736	147.356	1.896	147.55	2.056	147.88	2.216	148.28	2.376	147.75
1.744	147.364	1.904	147.57	2.064	147.9	2.224	148.26	2.384	147.72
1.752	147.372	1.912	147.58	2.072	147.93	2.232	148.25	2.392	147.69



13. 2D and 3D implication of 1D model

In this section we discuss the transformation of the ANSYS 1D adiabatic model to 2D/3D models simulating quenches in a magnet coil. As a reference we consider an SMC (Short Model Coil) designed at CERN to test new magnet technologies [39]. Figure 31 shows the ANSYS representation of an SMC.

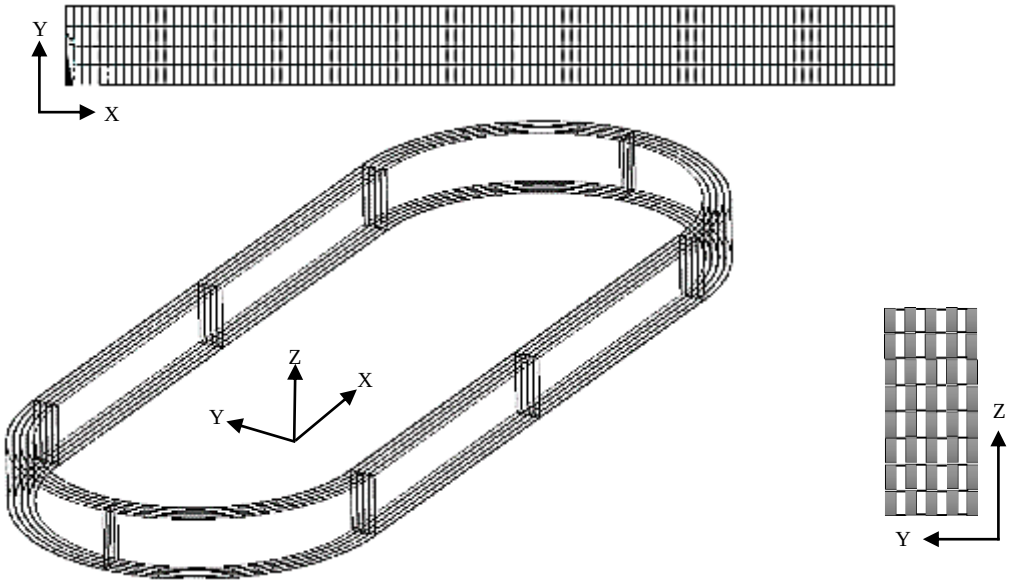


Figure 31 The geometry of the ANSYS model representing an SMC. Upper: 2D model, Lower left: 3D model and Lower right: cross-section view of the 3D model.

SMC consists of 21 - 35 layers of Rutherford-type cable each made up of 14 - 40 Nb_3Sn strands. The cables are insulated from one another with a thin layer of insulation material, S2-Glass.

The upper part of Figure 31 is a cut in the x-y plane through the straight section of an SMC. The horizontal line represents Rutherford-type cables. The cross-section area defined to the line is equivalent to the area of copper fraction in a cable. The vertical links in between the horizontal lines are 2 noded contact elements, "LINK33", representing insulation between the cables.

In the 3D model the cables are modelled using 8 noded solid elements, "SOLID70". To model insulation between the cables, we use contact elements, "LINK33", in between the solid elements as shown in lower right corner of Figure 31. The length of the coil, total number of cables and the scale of longitudinal discretization are taken as model parameters.

The material properties, magnetic field, current and other setting assigned to the models is in the same way as was done for 1D strand model. The APDL scripts to build 2D and 3D quench simulation models representing the SMC can be found in Appendices 18 and 19 respectively.

14. APDL scripts: Adiabatic model

```
finish
/clear
/prep7
/nerr,5
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Input Parameters !!!!!!!!!!!!!!!!!!!!!!!!!!!!
length = 1
toper = 1.9
tpick= 20
tpickloc = 0
b = 2.88
iop = 150
acu = 0.5547e-6
dcu = 0.8404e-3
rrr = 150
fcu = 0.62263
division = 1000
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
bc20 = 14.5
tco = 9.2
c1 = 3449
c2 = -257
tc = tco*(1-b/bc20)**0.59
tcs = tc*(1-iop/(c1+c2*b))
rhocu = 8960
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Material Property tables !!!!!!!!!!!!!!!!!!!
tmax = 100
*dim,isc,table,tmax,1,1,temp
*dim,rescu,table,tmax,1,1,temp
*dim,kcu,table,tmax,1,1,temp
*dim,cpcu,table,tmax,1,1,temp
*dim,cpnbti,table,tmax,1,1,temp
*dim,cpwire,table,tmax,1,1,temp
*dim,heatgen,table,tmax,1,1,temp
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Current in superconductor !!!!!!!!!!!!!!!!!!!!!
isc(0,1) = 0.0
*do,i,1,tmax,1
  isc(i,0) = i
  *if,i,lt,tcs,then
     isc(i,1) = iop
  *elseif,tcs,le,i,and,i,lt,tc,then
     isc(i,1) = iop*(1-(i-tcs)/(tc-tcs))
  *else
     isc(i,1) = 0
  *endif
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Resistivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!
a0 = 1.7
a1 = 2.33e9
a2 = 9.57e5
a3 = 163
rescu(0,1) = 0.0
*do,i,1,tmax,1
  rescu(i,0) = i
  rescu(i,1) = (a0/rrr+1/(a1/i**5+a2/i**3+a3/i))*10**(-8)+(0.37+0.0005*rrr)*b*10**(-10)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Thermal Conductivity !!!!!!!!!!!!!!!!!!!!
kcu(0,1) = 0.0
*do,i,1,tmax,1
```

```
  kcu(i,0) = i
  kcu(i,1) = 2.45*10**(-8)*i/rescu(i,1)
*enddo

!!!!!!!!!!!!!!!!!!!!!!!!!!!! Heat Generation !!!!!!!!!!!!!!!!!!!!!!!!!!!!
heatgen(0,1) = 0.0
*do,i,1,tmax,1
  heatgen(i,0) = i
  heatgen(i,1) = rescu(i,1)*((iop-isc(i,1))**2)/acu**2
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!
cpcu(0,1) = 0.0
*do,i,1,tmax,1
  cpcu(i,0) = i
  *if,i,lt,10,then
    a = (-3.080E-02)*i**4
    b = (7.230E+00)*i**3
    c = (-2.129E+00)*i**2
    d = (1.019E+02)*i
    e = 2.563E+00
  *elseif,10,le,i,and,i,lt,40,then
    a = (-3.045E-01)*i**4
    b =  (2.987E+01)*i**3
    c = (-4.556E+02)*i**2
    d =  (3.470E+03)*i
    e = -8.250E+03
  *elseif,40,le,i,and,i,lt,125,then
    a = (4.190E-02)*i**4
    b = (-1.402E+01)*i**3
    c = (1.509E+03)*i**2
    d = (-3.160E+04)*i
    e = 1.784E+05
  *elseif,125,le,i,and,i,lt,300,then
    a = (-8.480E-04)*i**4
    b = (8.419E-01)*i**3
    c = (-3.255E+02)*i**2
    d = (6.059E+04)*i
    e = -1.290E+06
  *elseif,300,le,i,and,i,lt,500,then
    a = (-4.800E-05)*i**4
    b = (9.173E-02)*i**3
    c = (-6.412E+01)*i**2
    d = (2.036E+04)*i
    e = 1.030E+06
  *else
    a = (0.000E+00)*i**4
    b = (1.200E-05)*i**3
    c = (-2.149E-01)*i**2
    d = (1.004E+03)*i
    e = 3.180E+06
  *endif
    cpcu(i,1) = a+b+c+d+e
  *enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! NbTi Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!
cpnbti(0,dt) = 0.0
*do,i,1,tmax,1
  cpnbti(i,0) = i
  *if,i,lt,tc,then
    a = (0.00000E+00)*i**4
    b = (4.91000E+01)*i**3
```

```
    c = (0.00000E+00)*i**2
    d = (6.40000E+01)*i
    e = 0.00000E+00
  *elseif,tc,le,i,and,i,lt,20,then
    a = (0.00000E+00)*i**4
    b =  (1.62400E+01)*i**3
    c = (0.00000E+00)*i**2
    d =  (9.28000E+02)*i
    e =  0.00000E+00
  *elseif,20,le,i,and,i,lt,50,then
    a = (-2.17700E-01)*i**4
    b = (1.19838E+01)*i**3
    c = (5.53710E+02)*i**2
    d = (-7.84610E+03)*i
    e = 4.13830E+04
  *elseif,50,le,i,and,i,lt,175,then
    a = (-4.82000E-03)*i**4
    b = (2.97600E+00)*i**3
    c = (-7.16300E+02)*i**2
    d = (8.30220E+04)*i
    e = -1.53000E+06
  *elseif,175,le,i,and,i,lt,500,then
    a = (-6.29000E-05)*i**4
    b = (9.29600E-02)*i**3
    c = (-5.16600E+01)*i**2
    d = (1.37060E+04)*i
    e = 1.24000E+06
  *else
    a = (0.00000E+00)*i**4
    b = (0.00000E+00)*i**3
    c = (-2.57000E-01)*i**2
    d = (9.55500E+02)*i
    e = 2.45000E+06
  *endif
    cpnbti(i,1) = a+b+c+d+e
*enddo
!!!!!!!!!!!!!!!!!!!!!! Equivalent Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpwire(0,1) = 0.0
*do,i,1,tmax,1
  cpwire(i,0) = i
  cpwire(i,1) = (cpcu(i,1)+(1/fcu-1)*cpnbti(i,1))/rhocu
*enddo
!!!!!!!!!!!!!!!!!!!!!! Nodes and Elements  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
npoints  = division+1
et,1,fluid116,1
r,1,dcu/2,acu
mp,dens,1,rhocu
mp,c,1,%cpwire%
mp,kxx,1,%kcu%
*do,i,1,npoints,1
  n,i,(i-1)*length/division,0,0
*enddo
type,1
mat,1
real,1
*do,i,1,division,1
  e,i,i+1
*enddo
Finish
!!!!!!!!!!!!!!!!!!!!!! Transient Anlysis setting !!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

```
/config,noeldb,0
/solu
antype,4
trnopt,full
kbc,1
eqslv,sparse
bcsoption,,default
lumpm,0
time,0.182
autots,on
solcontrol,on,on
deltim,1e-5,1e-5,1e-3
neqit,1000
lnsrch,on
rescontrol,define,none,none,1
!!!!!!!!!!!!!!!!! Initial condition, Loads, Boundary Conditions !!!!!!!!!!!!!!
*do,j,1,npoints,1
  ic,j,temp,toper+(tpick-toper)*exp(-(((j-1)*length/division-tpickloc)/0.099)**2)
*enddo
nsel,all
bf,all,hgen,%heatgen%
solve
finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Post Analysis !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/post1
nsel,s,loc,y,0
*dim,graph,table,npoints,2,1
*vget,graph(1,1),node,all,loc,x
*vget,graph(1,2),node,all,temp,y
*vplot,graph(1,1),graph(1,2)
/axlab,x,length
/axlab,y,Temperature
/replot
*cfopen,Temp_s,txt
*vwrite,graph(1,2)
(e20.5)
*cfclose
!plnsol,temp
!antime,50,0.1
Finish
```

15. APDL scripts: Helium cooled model (Model 2)

```
finish
/clear
/prep7
/nerr,5
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Input Parameters !!!!!!!!!!!!!!!!!!!!!!!!!!!!
length = 1
toper = 1.9
tpick= 20
tlamda = 2.1652
tpickloc = 0
b = 2.88
iop = 150
acu = 0.5547e-6
ahe = 1.0e-6
vhe = 1.0e-6*length
ahf = 1320e-6
dcu = sqrt(4*acu/3.1416)
rrr = 150
fcu = 0.62263
division = 1000
maxtime  = 0.01
nstep    = 100
substep  = 1
nkap     = 3
akap     = 200
afbii    = 200
anc      = 500
anb      = 50000
afbi     = 220
ekap     = 35000
enc      = 10
enb      = 20000
svhe = vhe/division
sahf = ahf/division
sacu = dcu/2*3.1416*length/division
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
bc20 = 14.5
tco = 9.2
c1 = 3449
c2 = -257
tc = tco*(1-b/bc20)**0.59
tcs = tc*(1-iop/(c1+c2*b))
rhocu = 8960
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Material Property tables !!!!!!!!!!!!!!!!!!!
tmax = 100
dt = 0.1
*dim,isc,table,tmax/dt,1,1,temp
*dim,rescu,table,tmax/dt,1,1,temp
*dim,kcu,table,tmax/dt,1,1,temp
*dim,cpcu,table,tmax/dt,1,1,temp
*dim,cpnbti,table,tmax/dt,1,1,temp
*dim,cpwire,table,tmax/dt,1,1,temp
*dim,heatgen,table,tmax/dt,1,1,temp
*dim,cphe,table,tmax/dt,1,1,temp
*dim,rhohe,table,tmax/dt,1,1,temp
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Current in superconductor !!!!!!!!!!!!!!!!!!!!
isc(0,1) = 0.0
*do,i,1,tmax/dt,1
```

```
  t = i*dt
  isc(i,0) = t
  *if,t,lt,tcs,then
     isc(i,1) = iop
  *elseif,tcs,le,t,and,t,lt,tc,then
     isc(i,1) = iop*(1-(t-tcs)/(tc-tcs))
  *else
     isc(i,1) = 0
  *endif
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Resistivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!
a0 = 1.7
a1 = 2.33e9
a2 = 9.57e5
a3 = 163
rescu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  rescu(i,0) = t
  rescu(i,1) = (a0/rrr+1/(a1/t**5+a2/t**3+a3/t))*10**(-8)+(0.37+0.0005*rrr)*b*10**(-10)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Thermal Conductivity !!!!!!!!!!!!!!!!!!!!
kcu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  kcu(i,0) = t
  kcu(i,1) = 2.45*10**(-8)*t/rescu(t,1)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Heat Generation !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
heatgen(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  heatgen(i,0) = t
  heatgen(i,1) = rescu(t,1)*((iop-isc(t,1))**2)/acu**2
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpcu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpcu(i,0) = t
  *if,t,lt,10,then
    a = (-3.080E-02)*t**4
    b = (7.230E+00)*t**3
    c = (-2.129E+00)*t**2
    d = (1.019E+02)*t
    e = 2.563E+00
  *elseif,10,le,t,and,t,lt,40,then
    a = (-3.045E-01)*t**4
    b =  (2.987E+01)*t**3
    c = (-4.556E+02)*t**2
    d =  (3.470E+03)*t
    e = -8.250E+03
  *elseif,40,le,t,and,t,lt,125,then
    a = (4.190E-02)*t**4
    b = (-1.402E+01)*t**3
    c = (1.509E+03)*t**2
    d = (-3.160E+04)*t
    e = 1.784E+05
  *elseif,125,le,t,and,t,lt,300,then
    a = (-8.480E-04)*t**4
```

```
    b = (8.419E-01)*t**3
    c = (-3.255E+02)*t**2
    d = (6.059E+04)*t
    e = -1.290E+06
  *elseif,300,le,t,and,t,lt,500,then
    a = (-4.800E-05)*t**4
    b = (9.173E-02)*t**3
    c = (-6.412E+01)*t**2
    d = (2.036E+04)*t
    e = 1.030E+06
  *else
    a = (0.000E+00)*t**4
    b = (1.200E-05)*t**3
    c = (-2.149E-01)*t**2
    d = (1.004E+03)*t
    e = 3.180E+06
  *endif
    cpcu(i,1) = a+b+c+d+e
 *enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! NbTi Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!
cpnbti(0,dt) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpnbti(i,0) = t
  *if,t,lt,tc,then
    a = (0.00000E+00)*t**4
    b = (4.91000E+01)*t**3
    c = (0.00000E+00)*t**2
    d = (6.40000E+01)*t
    e = 0.00000E+00
  *elseif,tc,le,t,and,t,lt,20,then
    a = (0.00000E+00)*t**4
    b =  (1.62400E+01)*t**3
    c = (0.00000E+00)*t**2
    d =  (9.28000E+02)*t
    e =  0.00000E+00
  *elseif,20,le,t,and,t,lt,50,then
    a = (-2.17700E-01)*t**4
    b = (1.19838E+01)*t**3
    c = (5.53710E+02)*t**2
    d = (-7.84610E+03)*t
    e = 4.13830E+04
  *elseif,50,le,t,and,t,lt,175,then
    a = (-4.82000E-03)*t**4
    b = (2.97600E+00)*t**3
    c = (-7.16300E+02)*t**2
    d = (8.30220E+04)*t
    e = -1.53000E+06
  *elseif,175,le,t,and,t,lt,500,then
    a = (-6.29000E-05)*t**4
    b = (9.29600E-02)*t**3
    c = (-5.16600E+01)*t**2
    d = (1.37060E+04)*t
    e = 1.24000E+06
  *else
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (-2.57000E-01)*t**2
    d = (9.55500E+02)*t
    e = 2.45000E+06
```

```
  *endif
    cpnbti(i,1) = a+b+c+d+e
*enddo
!!!!!!!!!!!!!!!!!!!!!! Equivalent Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpwire(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpwire(i,0) = t
  cpwire(i,1) = (cpcu(t,1)+(1/fcu-1)*cpnbti(t,1))/rhocu
*enddo
!!!!!!!!!!!!!!!!!!!!!! Helium Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
*TREAD,cphe,'cphe','txt','G:\Projects\lhccm\QuSi_ANSYS\apdl_projects',0
!!!!!!!!!!!!!!!!!!!!!!!!! Helium Density !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
*TREAD,rhohe,'rhohe','txt','G:\Projects\lhccm\QuSi_ANSYS\apdl_projects',0
!!!!!!!!!!!!!!!!!!!!!! Nodes and Elements  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
npoints  = division+1
et,1,fluid116,1
et,2,mass71,,,0
et,3,link34,,3,3
r,1,dcu,acu
r,2,svhe
r,3,sahf,0,0
r,4,sahf/2,0,0
mp,dens,1,rhocu
mp,c,1,%cpwire%
mp,kxx,1,%kcu%
mp,dens,2,%rhohe%
mp,c,2,%cphe%
*do,i,1,npoints,1
    mp,hf,i+2,1e-2
*enddo
*do,j,1,2,1
*do,i,1,npoints,1
  n,i+(j-1)*npoints,(i-1)*length/division,(j-1)*length/10,0
*enddo
*enddo
type,1
mat,1
real,1
*do,i,1,division,1
   e,i,i+1
*enddo
type,2
mat,2
real,2
*do,i,1,npoints,1
    e,i+npoints
*enddo
type,3
real,4
mat,3
e,1,1+npoints
type,3
real,3
mat,3
*do,i,2,npoints,1
   mat,2+i
   e,i,i+npoints
*enddo
finish
```

```
!!!!!!!!!!!!!!!!!!!! Transient Anlysis setting !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/config,noeldb,0
/solu
antype,4
trnopt,full
kbc,1
eqslv,sparse
bcsoption,,default
lumpm,0
autots,off
nsubst,1
neqit,1000
lnsrch,on
rescontrol,define,none,none,1
!!!!!!!!!!!!!!!!!! Initial condition, Loads, Boundary Conditions !!!!!!!!!!!!!!
*do,j,1,npoints,1
  ic,j,temp,toper+(tpick-toper)*exp(-(((j-1)*length/division-tpickloc)/0.1)**2)
*enddo
nsel,s,node,,1+npoints,2*npoints
ic,all,temp,toper
esel,s,type,,1
bfe,all,hgen,,%heatgen%
*dim,kenergy,array,npoints
*dim,tcu,array,npoints
*dim,the,array,npoints
*dim,regno,array,npoints
*dim,ncenergy,array,npoints
*dim,nbenergy,array,npoints
*do,j,1,nstep,1
  time,maxtime*j/nstep
  esel,all
  nsel,all
  solve
  *vget,tcu(1),node,1,temp,val
  *vget,the(1),node,1+npoints,temp,val
  *do,i,1,npoints,1
    *if,the(i),lt,tlamda,then
        *if,regno(i),eq,3,then
          kenergy(i) = abs(afbii*(tcu(i)-the(i)))
        *else,
          kenergy(i) = abs(akap*(tcu(i)**nkap-the(i)**nkap))
        *endif
        *if,kenergy(i),lt,ekap,then
              regno(i) = 2
              mp,hf,i+2,abs(akap*(tcu(i)**nkap-the(i)**nkap))/(abs(tcu(i)-the(i))+0.001)+0.001
         *else
          regno(i) = 3
          mp,hf,i+2,afbii
         *endif
     *else
        ncenergy(i) = abs(anc*(tcu(i)-the(i)))
        nbenergy(i)= sign(1, tcu(i)-the(i))*abs(anb*(tcu(i)-the(i))**2.5)
        *if,ncenergy(i),lt,enc,then
          regno(i) = 4
          mp,hf,i+2,anc
        *elseif,nbenergy(i),lt,enb,then
          regno(i) = 5
          mp,hf,i+2,abs(anb*(tcu(i)-the(i))**1.5)+0.001
        *else
          regno(i) = 6
```

```
            mp,hf,i+2,afbi
          *endif
      *endif
  *enddo
*enddo
finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Post Analysis !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/post1
nsel,s,loc,y,0
*dim,graph,table,npoints,2,1
*vget,graph(1,1),node,all,loc,x
*vget,graph(1,2),node,all,temp,y
*vplot,graph(1,1),graph(1,2)
/axlab,x,length
/axlab,y,Temperature
/replot
*cfopen,Temp_s,txt
*vwrite,graph(1,2)
(e20.5)
*cfclose
!plnsol,temp
!antime,50,0.1
Finish
```

16. APDL scripts: Helium cooled model (Model 1)

```
!!!!!!!!!!!!!!!!!!!!!! Nodes and Elements  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
npoints  = division+1
et,1,fluid116,1
et,2,link31,,,1
et,3,link34,,3,3
r,1,dcu,acu
r,2,dhe,ahe
r,3,ahf/division,0.0,0.0            !FBII/NC/FBI/ET3
r,4,ahf/division,1.5,0.0            !NB/ET3
r,5,ahf/division/2,0.0,0.0
r,6,ahf/division/2,1.5,0.0
mp,dens,1,rhocu
mp,c,1,%cpwire%
mp,kxx,1,%kcu%
mp,dens,2,%rhohe%
mp,c,2,%cphe%
mp,kxx,2,1e-6
mp,emis,3,1                         !KAP/ET2/R3/M3
mp,hf,4,afbii                       !FBII/ET3/R4/M4
mp,hf,5,anc                         !NC/ET3/R4/M5
mp,hf,6,anb                         !NB/ET3/R5/M6
mp,hf,7,afbi                        !FBI/ET3/R4/M7
*do,i,1,npoints,1
  n,i,(i-1)*length/division,0,0
*enddo
*do,i,1,npoints,1
  n,i+npoints,(i-1)*length/division,lhecu,0
*enddo
type,1
mat,1
real,1
*do,i,1,division,1
  e,i,i+1
*enddo
type,1
mat,2
real,2
*do,i,1,division,1
  e,i+npoints,i+npoints+1
*enddo
type,2
mat,3
*do,i,1,npoints,1
  *if,i,eq,1,then
  r,6+i,ahf/division/2,ahf/division/2,1,akap
  *else
  r,6+i,ahf/division,ahf/division,1,akap
  *endif
  real,6+i
  e,i,i+npoints
*enddo
type,3
real,5
mat,4
e,1,1+npoints
type,3
real,3
mat,4
```

```
*do,i,2,npoints,1
  e,i,i+npoints
*enddo
type,3
real,5
mat,5
e,1,1+npoints
type,3
real,3
mat,5
*do,i,2,npoints,1
  e,i,i+npoints
*enddo
type,3
real,6
mat,6
e,1,1+npoints
type,3
real,4
mat,6
*do,i,2,npoints,1
  e,i,i+npoints
*enddo
type,3
real,5
mat,7
e,1,1+npoints
type,3
real,3
mat,7
*do,i,2,npoints,1
  e,i,i+npoints
*enddo
Finish
!!!!!!!!!!!!!!!!!!!! Transient Anlysis setting !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/config,noeldb,0
/solu
antype,4
trnopt,full
kbc,1
eqslv,sparse
bcsoption,,default
lumpm,0
autots,off
solcontrol,on
nsubst,substep
neqit,1000
lnsrch,on
rescontrol,define,none,none,1
!!!!!!!!!!!!!!!!!! Initial condition, Loads, Boundary Conditions !!!!!!!!!!!!!!
*do,j,1,npoints,1
  ic,j,temp,toper+(tpick-toper)*exp(-(((j-1)*length/division-tpickloc)/0.099)**2)
*enddo
*do,i,1,npoints,1
  ic,i+npoints,temp,toper
*enddo
*do,i,1,npoints,1
  bf,i,hgen,%heatgen%
*enddo
*do,i,1,4*npoints,1
```

```
  bfe,i+2*division+npoints,hgen,,0
*enddo
*dim,kenergy,array,npoints
*dim,ncenergy,array,npoints
*dim,nbenergy,array,npoints
*dim,tcu,array,npoints
*dim,the,array,npoints
*dim,regno,array,npoints
esel,s,elem,,1+npoints+2*division,2*division+5*npoints,1
ekill,all
*do,j,1,nstep,1
  time,maxtime*j/nstep
  esel,all
  solve
  *do,i,1,npoints,1
    *get,tcu(i),node,i,temp
    *get,the(i),node,i+npoints,temp
  *enddo
  esel,s,elem,,1+2*division,2*division+5*npoints,1
  ekill,all
  *do,i,1,npoints,1
    *if,the(i),lt,tlamda,then
      *if,regno(i),eq,3,then
      kenergy(i)= abs(afbii*(tcu(i)-the(i)))
      *else
      kenergy(i)= abs(akap*(tcu(i)**nkap-the(i)**nkap))
      *endif
      *if,kenergy(i),lt,ekap,then
        aakap = ahf/division*the(i)**(nkap-4)
        fkap  = ahf/division*tcu(i)**(nkap-4)
        regno(i) = 2
        esel,s,elem,,i+2*division
        ealive,all
        *if,i,eq,1,then
        rmodif,6+i,1,aakap/2,fkap/2
        *else
        rmodif,6+i,1,aakap,fkap
        *endif
      *else
        regno(i) = 3
        esel,s,elem,,i+2*division+npoints
        ealive,all
      *endif
    *else
       ncenergy(i) = abs(anc*(tcu(i)-the(i)))
       nbenergy(i)= sign(1, tcu(i)-the(i))*abs(anb*(tcu(i)-the(i))**2.5)
       *if,ncenergy(i),lt,enc,then
         regno(i) = 4
         esel,s,elem,,i+2*division+2*npoints
         ealive,all
       *elseif,nbenergy(i),lt,enb,then
          regno(i) = 5
          esel,s,elem,,i+2*division+3*npoints
          ealive,all
       *else
          regno(i) = 6
          esel,s,elem,,i+2*division+4*npoints
          ealive,all
       *endif
    *endif
```

```
  *enddo
*enddo
finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Post Analysis !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/post1
nsel,s,loc,y,0
*dim,graph,table,npoints,2,1
*vget,graph(1,1),node,all,loc,x
*vget,graph(1,2),node,all,temp,y
*vplot,graph(1,1),graph(1,2)
/axlab,x,length
/axlab,y,Temperature
/replot
*cfopen,Temp_s,txt
*vwrite,graph(1,2)
(e20.5)
*cfclose
!plnsol,temp
!antime,50,0.1
Finish
```

17. APDL scripts: Helium cooled model (Model 3)

```
!!!!!!!!!!!!!!!!!!!!!!! Nodes and Elements  !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
npoints  = division+1
et,1,fluid116,1
r,1,dcu,acu
mp,dens,1,rhocu
mp,c,1,%cpwire%
mp,kxx,1,%kcu%
*do,i,1,npoints,1
  n,i,(i-1)*length/division,0,0
*enddo
type,1
mat,1
real,1
*do,i,1,division,1
  e,i,i+1
*enddo
Finish
!!!!!!!!!!!!!!!!!!!!!! Transient Anlysis setting !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/config,noeldb,0
/solu
antype,4
trnopt,full
kbc,1
autots,off
nsubst,1
eqslv,sparse
bcsoption,,default
lumpm,0
neqit,1000
lnsrch,on
rescontrol,define,none,none,1
!!!!!!!!!!!!!!!!!! Initial condition, Loads, Boundary Conditions !!!!!!!!!!!!!!!
*do,j,1,npoints,1
  ic,j,temp,toper+(tpick-toper)*exp(-(((j-1)*length/division-tpickloc)/0.099)**2)
*enddo
*dim,kenergy,array,npoints
*dim,ncenergy,array,npoints
*dim,nbenergy,array,npoints
*dim,fbenergy,array,npoints
*dim,tcu,array,npoints
*dim,the,array,npoints
*dim,regno,array,npoints
*do,i,1,npoints,1
 the(i) = toper
*enddo
*do,j,1,2868,1
  time,timestep(j,1)
  esel,all
  solve
  *do,i,1,npoints-1,1
    *get,tcu(i),node,i,temp
    *if,regno(i),eq,4,then
     the(i) = the(i)+ncenergy(i)*sahf/(svhe*rhohe(the(i),1)*cphe(the(i),1))*(timestep(j,1)-timestep(j-1,1))
    *elseif,regno(i),eq,5,then
      the(i) = the(i)+nbenergy(i)*sahf/(svhe*rhohe(the(i),1)*cphe(the(i),1))*(timestep(j,1)-timestep(j-1,1))
    *elseif,regno(i),eq,6,then
      the(i) = the(i)+fbenergy(i)*sahf/(svhe*rhohe(the(i),1)*cphe(the(i),1))*(timestep(j,1)-timestep(j-1,1))
    *else
```

```
     the(i) = the(i)+kenergy(i)*sahf/(svhe*rhohe(the(i),1)*cphe(the(i),1))*(timestep(j,1)-timestep(j-1,1))
    *endif
  *enddo
  *do,i,1,npoints-1,1
    *if,the(i),lt,tlamda,then
     *if,regno(i),eq,3,then
     kenergy(i)= abs(afbii*(tcu(i)-the(i)))
     *else
     kenergy(i)= abs(akap*(tcu(i)**nkap-the(i)**nkap))
     *endif
     bfe,i,hgen,,heatgen(tcu(i))-kenergy(i)*sahf/(acu*length/division)
     *if,kenergy(i),lt,ekap,then
      regno(i) = 2
     *else
      regno(i) = 3
     *endif
    *else
      ncenergy(i) = abs(anc*(tcu(i)-the(i)))
      nbenergy(i)= sign(1, tcu(i)-the(i))*abs(anb*(tcu(i)-the(i))**2.5)
      *if,ncenergy(i),lt,enc,then
       regno(i) = 4
       bfe,i,hgen,,heatgen(tcu(i))-ncenergy(i)*sahf/(acu*length/division)
      *elseif,nbenergy(i),lt,enb,then
        regno(i) = 5
        bfe,i,hgen,,heatgen(tcu(i))-nbenergy(i)*sahf/(acu*length/division)
      *else
        regno(i) = 6
        fbenergy(i) = afbi*(tcu(i)-the(i))
        bfe,i,hgen,,heatgen(tcu(i))-fbenergy(i)*sahf/(acu*length/division)
      *endif
    *endif
  *enddo
*enddo
finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Post Analysis !!!!!!!!!!!!!!!!!!!!!!!!!!!!
/post1
nsel,s,loc,y,0
*dim,graph,table,npoints,2,1
*vget,graph(1,1),node,all,loc,x
*vget,graph(1,2),node,all,temp,y
*vplot,graph(1,1),graph(1,2)
/axlab,x,length
/axlab,y,Temperature
/replot
*cfopen,Temp_s,txt
*vwrite,graph(1,2)
(e20.5)
*cfclose
!plnsol,temp
!antime,50,0.1
Finish
```

18. APDL scripts: 2D quench simulation model of a magnet coil

```
finish
/clear,all
/prep7
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Input Values !!!!!!!!!!!!!!!!!!!!!!!!!!!
length = 1
ruthno = 5
ttf    = 2
lhecu  = 240*10**(-(6-ttf))
hcu    = 15e-3
wcu    = 2e-3
division = 100
maxtime  = 0.2
nstep    = 1
substep  = 1000
toper = 1.9
tpick = 20
tpickloc = 0
b = 2.88
iop = 150
rrr = 150
fcu = 0.62263
acu = 0.5547e-6
dcu = 0.8404e-3
hcu    = dcu*14
wcu    = dcu*2
ahf    = hcu*length/(division+1)
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Derived Parameters !!!!!!!!!!!!!!!!!!!!!!!!!!!
bc20 = 14.5
tco  = 9.2
c1 = 3449
c2 = -257
tc = tco*(1-b/bc20)**0.59
tcs = tc*(1-iop/(c1+c2*b))
rhocu = 8960
rhokap = 1420
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Material Property tables !!!!!!!!!!!!!!!!!!!
tmax = 100
dt = 0.1
*dim,isc,table,tmax/dt,1,1,temp
*dim,rescu,table,tmax/dt,1,1,temp
*dim,kcu,table,tmax/dt,1,1,temp
*dim,cpcu,table,tmax/dt,1,1,temp
*dim,cpnb3sn,table,tmax/dt,1,1,temp
*dim,cpwire,table,tmax/dt,1,1,temp
*dim,kg10pll,table,tmax/dt,1,1,temp
*dim,kg10nl,table,tmax/dt,1,1,temp
*dim,cpg10,table,tmax/dt,1,1,temp
*dim,heatgen,table,tmax/dt,1,1,temp
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Current in superconductor !!!!!!!!!!!!!!!!!!!!
isc(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  isc(i,0) = t
  *if,t,lt,tcs,then
     isc(i,1) = iop
  *elseif,tcs,le,t,and,t,lt,tc,then
     isc(i,1) = iop*(1-(t-tcs)/(tc-tcs))
  *else
```

```
    isc(i,1) = 0
  *endif
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Resistivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!
a0 = 1.7
a1 = 2.33e9
a2 = 9.57e5
a3 = 163
rescu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  rescu(i,0) = t
  rescu(i,1) = (a0/rrr+1/(a1/t**5+a2/t**3+a3/t))*10**(-8)+(0.37+0.0005*rrr)*b*10**(-10)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Thermal Conductivity !!!!!!!!!!!!!!!!!!!!
kcu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  kcu(i,0) = t
  kcu(i,1) = 2.45*10**(-8)*t/rescu(t,1)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Heat Generation !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
heatgen(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  heatgen(i,0) = t
  heatgen(i,1) = rescu(t,1)*((iop-isc(t,1))**2)/acu**2
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!
cpcu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpcu(i,0) = t
  *if,t,lt,10,then
    a = (-3.080E-02)*t**4
    b = (7.230E+00)*t**3
    c = (-2.129E+00)*t**2
    d = (1.019E+02)*t
    e = 2.563E+00
  *elseif,10,le,t,and,t,lt,40,then
    a = (-3.045E-01)*t**4
    b =  (2.987E+01)*t**3
    c = (-4.556E+02)*t**2
    d =  (3.470E+03)*t
    e = -8.250E+03
  *elseif,40,le,t,and,t,lt,125,then
    a = (4.190E-02)*t**4
    b = (-1.402E+01)*t**3
    c = (1.509E+03)*t**2
    d = (-3.160E+04)*t
    e = 1.784E+05

  *elseif,125,le,t,and,t,lt,300,then
    a = (-8.480E-04)*t**4
    b = (8.419E-01)*t**3
    c = (-3.255E+02)*t**2
    d = (6.059E+04)*t
    e = -1.290E+06
  *elseif,300,le,t,and,t,lt,500,then
    a = (-4.800E-05)*t**4
```

```
    b = (9.173E-02)*t**3
    c = (-6.412E+01)*t**2
    d = (2.036E+04)*t
    e = 1.030E+06
  *else
    a = (0.000E+00)*t**4
    b = (1.200E-05)*t**3
    c = (-2.149E-01)*t**2
    d = (1.004E+03)*t
    e = 3.180E+06
  *endif
    cpcu(i,1) = a+b+c+d+e
  *enddo
!!!!!!!!!!!!!!!!!!!!!! NB3Sn Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!
cpnb3sn(0,dt) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpnb3sn(i,0) = t
  *if,t,lt,tc,then
    a = (0.00000E+00)*t**4
    b = (38.8 - 1.8*bh+0.0634*bh**2)*t**3
    c = (-110*10**(-0.434*bh))*t**2
    d = (207-3.83*bh+2.86*bh**2)*t
    e = 0.00000E+00
  *elseif,tc,le,t,and,t,lt,26.113,then
    a = (0.00000E+00)*t**4
    b =  (7.4200E+00)*t**3
    c = (0.00000E+00)*t**2
    d =  (1.522000E+03)*t
    e =  0.00000E+00
  *elseif,26.113,le,t,and,t,lt,169.416,then
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (-6.16350E+01)*t**2
    d = (1.9902E+04)*t
    e = -3.058070E+05
  *elseif,169.416,le,t,and,t,lt,300,then
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (-7.46360E+00)*t**2
    d = (4.411E+03)*t
    e = 7.638010E+05
  *else
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (0.000E+00)*t**2
    d = (0.0000E+00)*t
    e = 1.415377E+06
  *endif
    cpnb3sn(i,1) = a+b+c+d+e
*enddo
!!!!!!!!!!!!!!!!!!!!!! Equivalent Heat Capacity 2 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpwire(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpwire(i,0) = t
  cpwire(i,1) = (cpcu(t,1)+(1/fcu2-1)*cpnb3sn(t,1))/rhocu
*enddo
!!!!!!!!!!!!!!!!!!!!!! G10 Normal Thermal Conductivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
kg10nl(0,1) = 0.0
```

```
*do,i,1,tmax/dt,1
  t = i*dt
  kg10nl(i,0) = t
    a = -4.1236
    b = 13.788*log10(t)
    c = -26.068*log10(t)**2
    d = 26.272*log10(t)**3
    e = -14.663*log10(t)**4
    f = 4.4954*log10(t)**5
    g = -0.6975*log10(t)**6
    h = 0.0397*log10(t)**7
    kg10nl(i,1) = 10**(a+b+c+d+e+f+g+h)
*enddo
!!!!!!!!!!!!!!!!!!!! G10 Parallel Thermal Conductivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
kg10pll(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  kg10pll(i,0) = t
    a = -2.64827
    b = 8.80228*log10(t)
    c = -24.8998*log10(t)**2
    d = 41.1625*log10(t)**3
    e = -39.8754*log10(t)**4
    f = 23.1778*log10(t)**5
    g = -7.95635*log10(t)**6
    h = 1.48806*log10(t)**7
    h1 = -0.11701*log10(t)**8
    kg10pll(i,1) = 10**(a+b+c+d+e+f+g+h)
*enddo
!!!!!!!!!!!!!!!!!!!! G10 Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpg10(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpg10(i,0) = t
    a = -2.4083
    b = 7.6006*log10(t)
    c = -8.2982*log10(t)**2
    d = 7.3301*log10(t)**3
    e = -4.2386*log10(t)**4
    f = 1.4294*log10(t)**5
    g = -0.24396*log10(t)**6
    h = 0.015236*log10(t)**7
  cpg10(i,1) = 10**(a+b+c+d+e+f+g+h)
*enddo
!!!!!!!!!!!!!!!!!!!! Element types and Material types !!!!!!!!!!!!!!!!!!
npoints = division + 1
et,1,fluid116,1
et,2,link33
r,1,,acu*28
r,2,ahf
mp,dens,1,rhocu
mp,dens,2,rhokap
mptgen,1,90,dt,dt
mptgen,91,10,10,10
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,kxx,1,i,kcu(i*dt,1)
 *else
  mpdata,kxx,1,i,kcu((i-90)*10,1)
 *endif
```

```
*enddo
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,c,1,i,cpwire(i*dt,1)
 *else
  mpdata,c,1,i,cpwire((i-90)*10,1)
 *endif
*enddo
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,kxx,2,i,kg10n1(i*dt,1)*10**ttf
 *else
  mpdata,kxx,2,i,kg10nl((i-90)*10,1)*10**ttf
 *endif
*enddo
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,c,2,i,cpg10(i*dt,1)*10**(-ttf)
 *else
  mpdata,c,2,i,cpg10((i-90)*10,1)*10**(-ttf)
 *endif
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Nodes and Elements !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
*do,j,1,ruthno,1
*do,i,1,npoints,1
  n,i+(j-1)*npoints,(i-1)*length/division,(j-1)*lhecu,0
*enddo
*enddo
type,1
mat,1
real,1
*do,j,1,ruthno,1
*do,i,1,division,1
  e,i+(j-1)*npoints,i+(j-1)*npoints+1
*enddo
*enddo
type,2
mat,2
real,2
*do,j,1,ruthno-1,1
*do,i,1,npoints,1
  e,i+(j-1)*npoints,i+j*npoints
*enddo
*enddo
finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Transient Analysis Setting !!!!!!!!!!!!!!!!!!!!!
/solu
antype,4
trnopt,full
kbc,0
eqslv,sparse
bcsoption,,default
lumpm,0
autots,on
time,maxtime
nsubst,substep
neqit,1000
lnsrch,on
outres,all,all
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Initial Condition, Loads, Boundary Conditions
```

```
*do,j,1,npoints,1
  ic,j,temp,toper+(tpick-toper)*exp(-(((j-1)*length/division-tpickloc)/0.0143)**2)
*enddo
nsel,s,node,,1+npoints,6*npoints,1
ic,all,temp,toper
nsel,all
bf,all,hgen,%heatgen%
esel,s,type,,2
bfe,all,hgen,,0
esel,all
solve
finish
/post1
nsel,s,loc,y,0
*dim,graph,table,6*npoints,2,1
*vget,graph(1,1),node,all,loc,x
*vget,graph(1,2),node,all,temp,y
*vplot,graph(1,1),graph(1,2)
/axlab,x,length
/axlab,y,Temperature
/replot
*cfopen,Temp_s,txt
*vwrite,graph(1,2)
(e20.5)
*cfclose
!plnsol,temp
!antime,50,0.1
!finish
```

19. APDL scripts: 3D quench simulation model of a magnet coil

```
finish
/clear,all
/prep7
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Input Values !!!!!!!!!!!!!!!!!!!!!!!!!!!
length = 1
lhecu  = 240e-6
division = 100
ruthno   = 5
maxtime  = 0.2
toper = 1.9
tpick = 20
tpickloc = 0
bh = 2.88
iop = 150
rrr = 150
fcu = 0.62263
fcu2 = 1.25/2.25
acu = 0.5547e-6
dcu = 0.8404e-3
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Derived parameters !!!!!!!!!!!!!!!!!!!!!!
ahf  = 14*sqrt(acu)*length/(15*(division+1))
bc20 = 14.5
tco  = 9.2
c1 = 3449
c2 = -257
tc = tco*(1-bh/bc20)**0.59
tcs = tc*(1-iop/(c1+c2*b))
rhocu = 8960
rhokap = 1420
rhog10 = 1760
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Material Property tables !!!!!!!!!!!!!!!!!!!
tmax = 100
dt = 0.1
*dim,isc,table,tmax/dt,1,1,temp
*dim,rescu,table,tmax/dt,1,1,temp
*dim,kcu,table,tmax/dt,1,1,temp
*dim,cpcu,table,tmax/dt,1,1,temp
*dim,cpnbti,table,tmax/dt,1,1,temp
*dim,cpwire,table,tmax/dt,1,1,temp
*dim,kkap,table,tmax/dt,1,1,temp
*dim,cpkap,table,tmax/dt,1,1,temp
*dim,cpnb3sn,table,tmax/dt,1,1,temp
*dim,cpwire2,table,tmax/dt,1,1,temp
*dim,cpg10,table,tmax/dt,1,1,temp
*dim,kg10nl,table,tmax/dt,1,1,temp
*dim,kg10pll,table,tmax/dt,1,1,temp
*dim,heatgen,table,tmax/dt,1,1,temp
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Current in superconductor !!!!!!!!!!!!!!!!!!!!!
isc(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  isc(i,0) = t
  *if,t,lt,tcs,then
     isc(i,1) = iop
  *elseif,tcs,le,t,and,t,lt,tc,then
     isc(i,1) = iop*(1-(t-tcs)/(tc-tcs))
  *else
     isc(i,1) = 0
```

```
  *endif
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Resistivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!
a0 = 1.7
a1 = 2.33e9
a2 = 9.57e5
a3 = 163
rescu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  rescu(i,0) = t
  rescu(i,1) = (a0/rrr+1/(a1/t**5+a2/t**3+a3/t))*10**(-8)+(0.37+0.0005*rrr)*bh*10**(-10)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Thermal Conductivity !!!!!!!!!!!!!!!!!!!!
kcu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  kcu(i,0) = t
  kcu(i,1) = 2.45*10**(-8)*t/rescu(t,1)
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Heat Generation !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
heatgen(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  heatgen(i,0) = t
  heatgen(i,1) = rescu(t,1)*((iop-isc(t,1))**2)/acu**2
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!! Copper Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpcu(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpcu(i,0) = t
  *if,t,lt,10,then
    a = (-3.080E-02)*t**4
    b = (7.230E+00)*t**3
    c = (-2.129E+00)*t**2
    d = (1.019E+02)*t
    e = 2.563E+00
  *elseif,10,le,t,and,t,lt,40,then
    a = (-3.045E-01)*t**4
    b =  (2.987E+01)*t**3
    c = (-4.556E+02)*t**2
    d =  (3.470E+03)*t
    e = -8.250E+03
  *elseif,40,le,t,and,t,lt,125,then
    a = (4.190E-02)*t**4
    b = (-1.402E+01)*t**3
    c = (1.509E+03)*t**2
    d = (-3.160E+04)*t
    e = 1.784E+05
  *elseif,125,le,t,and,t,lt,300,then
    a = (-8.480E-04)*t**4
    b = (8.419E-01)*t**3
    c = (-3.255E+02)*t**2
    d = (6.059E+04)*t
    e = -1.290E+06
  *elseif,300,le,t,and,t,lt,500,then
    a = (-4.800E-05)*t**4
    b = (9.173E-02)*t**3
    c = (-6.412E+01)*t**2
```

```
    d = (2.036E+04)*t
    e = 1.030E+06
  *else
    a = (0.000E+00)*t**4
    b = (1.200E-05)*t**3
    c = (-2.149E-01)*t**2
    d = (1.004E+03)*t
    e = 3.180E+06
  *endif
    cpcu(i,1) = a+b+c+d+e
  *enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! NbTi Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!
cpnbti(0,dt) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpnbti(i,0) = t
  *if,t,lt,tc,then
    a = (0.00000E+00)*t**4
    b = (4.91000E+01)*t**3
    c = (0.00000E+00)*t**2
    d = (6.40000E+01)*t
    e = 0.00000E+00
  *elseif,tc,le,t,and,t,lt,20,then
    a = (0.00000E+00)*t**4
    b =  (1.62400E+01)*t**3
    c = (0.00000E+00)*t**2
    d =  (9.28000E+02)*t
    e =  0.00000E+00
  *elseif,20,le,t,and,t,lt,50,then
    a = (-2.17700E-01)*t**4
    b = (1.19838E+01)*t**3
    c = (5.53710E+02)*t**2
    d = (-7.84610E+03)*t
    e = 4.13830E+04
  *elseif,50,le,t,and,t,lt,175,then
    a = (-4.82000E-03)*t**4
    b = (2.97600E+00)*t**3
    c = (-7.16300E+02)*t**2
    d = (8.30220E+04)*t
    e = -1.53000E+06
  *elseif,175,le,t,and,t,lt,500,then
    a = (-6.29000E-05)*t**4
    b = (9.29600E-02)*t**3
    c = (-5.16600E+01)*t**2
    d = (1.37060E+04)*t
    e = 1.24000E+06
  *else
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (-2.57000E-01)*t**2
    d = (9.55500E+02)*t
    e = 2.45000E+06
  *endif
    cpnbti(i,1) = a+b+c+d+e
*enddo
!!!!!!!!!!!!!!!!!!!!!!! NB3Sn Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!
cpnb3sn(0,dt) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpnb3sn(i,0) = t
```

```
  *if,t,lt,tc,then
    a = (0.00000E+00)*t**4
    b = (38.8 - 1.8*bh+0.0634*bh**2)*t**3
    c = (-110*10**(-0.434*bh))*t**2
    d = (207-3.83*bh+2.86*bh**2)*t
    e = 0.00000E+00
  *elseif,tc,le,t,and,t,lt,26.113,then
    a = (0.00000E+00)*t**4
    b =  (7.4200E+00)*t**3
    c = (0.00000E+00)*t**2
    d =  (1.522000E+03)*t
    e =  0.00000E+00
  *elseif,26.113,le,t,and,t,lt,169.416,then
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (-6.16350E+01)*t**2
    d = (1.9902E+04)*t
    e = -3.058070E+05
  *elseif,169.416,le,t,and,t,lt,300,then
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (-7.46360E+00)*t**2
    d = (4.411E+03)*t
    e = 7.638010E+05
  *else
    a = (0.00000E+00)*t**4
    b = (0.00000E+00)*t**3
    c = (0.000E+00)*t**2
    d = (0.0000E+00)*t
    e = 1.415377E+06
  *endif
    cpnb3sn(i,1) = a+b+c+d+e
*enddo
!!!!!!!!!!!!!!!!!!!!!! Equivalent Heat Capacity 2 !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpwire(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpwire2(i,0) = t
  cpwire(i,1) = (cpcu(t,1)+(1/fcu2-1)*cpnb3sn(t,1))/rhocu
*enddo
!!!!!!!!!!!!!!!!!!!!!! G10 Normal Thermal Conductivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
kg10nl(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  kg10nl(i,0) = t
    a = -4.1236
    b = 13.788*log10(t)
    c = -26.068*log10(t)**2
    d = 26.272*log10(t)**3
    e = -14.663*log10(t)**4
    f = 4.4954*log10(t)**5
    g = -0.6975*log10(t)**6
    h = 0.0397*log10(t)**7
    kg10nl(i,1) = 10**(a+b+c+d+e+f+g+h)
*enddo
!!!!!!!!!!!!!!!!!!!!!! G10 Parallel Thermal Conductivity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!
kg10pll(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  kg10pll(i,0) = t
```

```
   a = -2.64827
   b = 8.80228*log10(t)
   c = -24.8998*log10(t)**2
   d = 41.1625*log10(t)**3
   e = -39.8754*log10(t)**4
   f = 23.1778*log10(t)**5
   g = -7.95635*log10(t)**6
   h = 1.48806*log10(t)**7
   h1 = -0.11701*log10(t)**8
   kg10pll(i,1) = 10**(a+b+c+d+e+f+g+h)
*enddo
!!!!!!!!!!!!!!!!!!!! G10 Heat Capacity !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
cpg10(0,1) = 0.0
*do,i,1,tmax/dt,1
  t = i*dt
  cpg10(i,0) = t
   a = -2.4083
   b = 7.6006*log10(t)
   c = -8.2982*log10(t)**2
   d = 7.3301*log10(t)**3
   e = -4.2386*log10(t)**4
   f = 1.4294*log10(t)**5
   g = -0.24396*log10(t)**6
   h = 0.015236*log10(t)**7
  cpg10(i,1) = 10**(a+b+c+d+e+f+g+h)
*enddo
!!!!!!!!!!!!!!!!!!!! Element types and Material types !!!!!!!!!!!!!!!!!!!!!!!!
npoints = division + 1
et,1,solid70
et,2,link33
mp,dens,1,rhocu
mp,dens,2,rhokap
mptgen,1,90,dt,dt
mptgen,91,10,10,10
!!!!!!!!!!!!!!!!!!!!!!!!!1 Nonlinear Material Property definition !!!!!!!!!!!1
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,kxx,1,i,kcu(i*dt,1)
 *else
  mpdata,kxx,1,i,kcu((i-90)*10,1)
 *endif
*enddo
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,c,1,i,cpwire(i*dt,1)
 *else
  mpdata,c,1,i,cpwire((i-90)*10,1)
 *endif
*enddo
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,kxx,2,i,kg10(i*dt,1)
 *else
  mpdata,kxx,2,i,kg10((i-90)*10,1)
 *endif
*enddo
*do,i,1,100,1
 *if,i,le,90,then
  mpdata,c,2,i,cpg10(i*dt,1)
 *else
```

```
  mpdata,c,2,i,cpg10((i-90)*10,1)
 *endif
*enddo
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Nodes and Elements !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
n,1, 0              ,0       ,0
n,2, length/division,0       ,0
n,3, length/division,sqrt(acu),0
n,4, 0              ,sqrt(acu),0
ngen,3,4,1,4,1,0,0,sqrt(acu)
type,1                        ! Elements type for superconductor
mat,1
e,1,2,3,4,5,6,7,8
e,5,6,7,8,9,10,11,12
esel,all
egen,ruthno,12,all,,,,,,,0,0,2*sqrt(acu)+lhecu
type,2                        ! Element type for Insulation
mat,2
r,1,ahf
r,2,ahf
r,3,ahf
r,4,ahf
!!!!!!!!!!!!!!! Generating insulation elements between the superconductor element
*do,i,1,ruthno-1,1
*do,j,1,4,1
  real,j
  e,j+8+(i-1)*12,j+(i-1)*12+12
*enddo
*enddo
esel,s,type,,1
esel,a,real,,3,4
egen,14,12*ruthno,all,,,,,,,0,sqrt(acu),0
esel,s,type,,1
esel,a,real,,2,3
egen,division,14*12*ruthno,all,,,,,,,,length/division,0,0
nummrg,node
finish
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Transient Analysis Setting !!!!!!!!!!!!!!!!!!!!
/solu
antype,4
trnopt,full
kbc,0
eqslv,sparse
bcsoption,,default
lumpm,0
autots,on
time,maxtime
deltime, 1e-5, 1e-5, 5e-4
neqit,1000
lnsrch,on
outres,all,all
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! Initial Condition, Loads, Boundary Conditions !!!!!
! Initial temperature profile
*do,i,1,npoints,1
nsel,s,loc,x,(i-1)*length/(division)
nsel,r,loc,y,0,5*sqrt(acu),sqrt(acu)
nsel,r,loc,z,0,2*sqrt(acu),sqrt(acu)
ic,all,temp,toper+(tpick-toper)*exp(-(((i-1)*length/division-tpickloc)/0.0141)**2)
*enddo
nsel,s,loc,z,0,2*sqrt(acu),sqrt(acu)
nsel,r,loc,y,0,5*sqrt(acu),sqrt(acu)
```

```
nsel,inve
ic,all,temp,toper
!Applying heat generation
esel,s,type,,1
bfe,all,hgen,,%heatgen%
esel,s,type,,2
bfe,all,hgen,,0
esel,all
nsel,all
solve
finish
/post1
!nsel,s,loc,y,0
!*dim,graph,table,6*npoints,2,1
!*vget,graph(1,1),node,all,loc,x
!*vget,graph(1,2),node,all,temp,y
!*vplot,graph(1,1),graph(1,2)
!/axlab,x,length
!/axlab,y,Temperature
!/replot
!*cfopen,Temp_s,txt
!*vwrite,graph(1,2)
!(e20.5)
!*cfclose
plnsol,temp
/contour,all,9,1.9,,tc     ! The grey color represent temperature above tc
/color,smax,14
/color,outl,0
!antime,50,0.1
!finish
```